# Opportunities and Challenges of More Flexible Networked Systems

**Stefan Schmid**

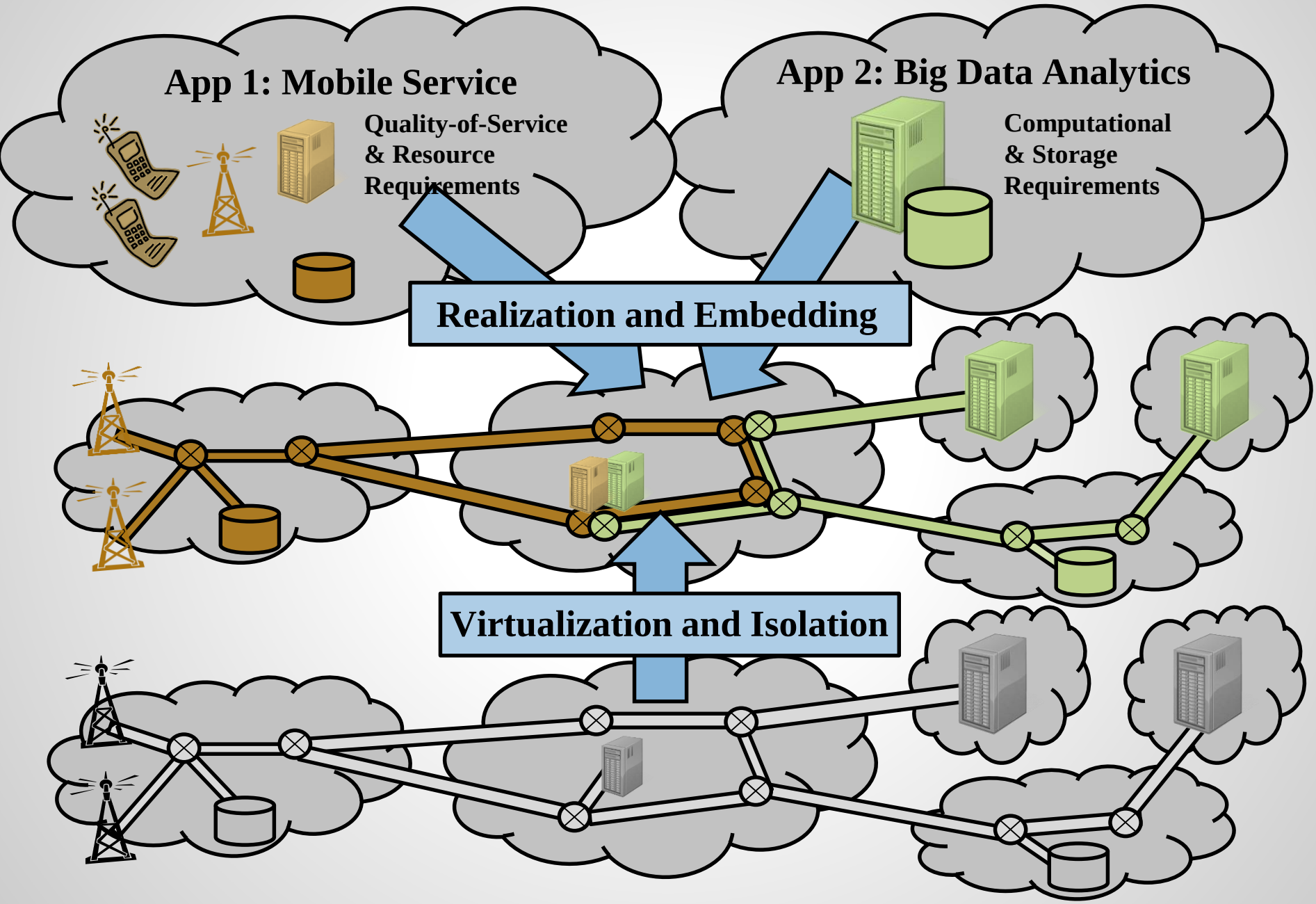
Aalborg University, Denmark

# Modern Networked Systems: Virtualized and Programmable

## New Flexibilities and Challenges


”We are at an interesting inflection point!”
Keynote by George Varghese
at SIGCOMM 2014

# Opportunity 1: Virtualization & Efficient Resource Sharing



# Opportunity 1: Virtualization & Efficient Resource Sharing

App 1

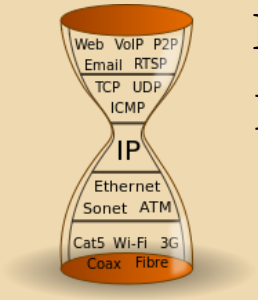
App 2: Big Data Analytics

**But also challenges:**

- ❑ Predictable performance: reservations across all resources. But how to allocate and schedule resources efficiently?
- ❑ How to provide true performance isolation?
- ❑ How to deal with imperfections / stragglers?
- ❑ How to exploit redundancies?

**Also:**

- ❑ How to tailor protocol stacks?
- ❑ Ossification at network core

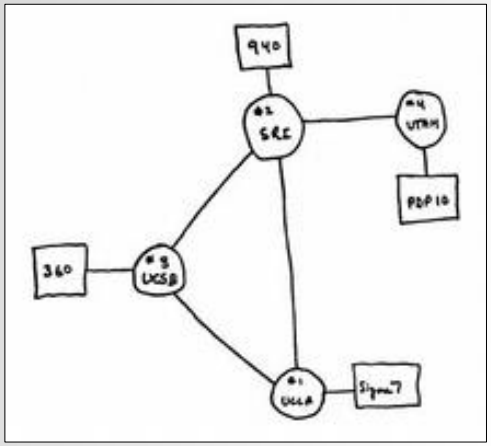


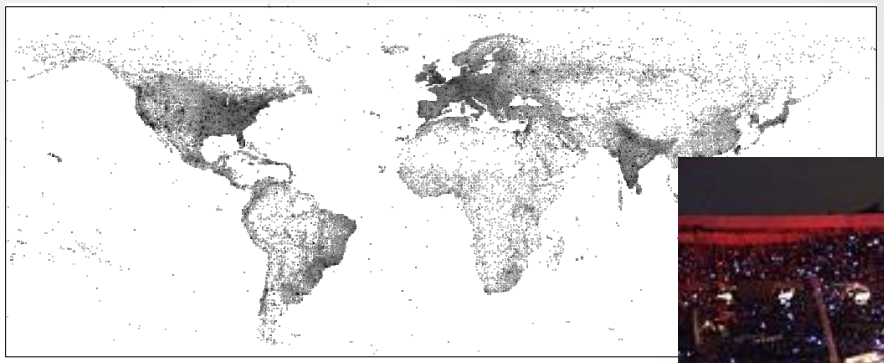

**Innovation!**

**Innovation??**

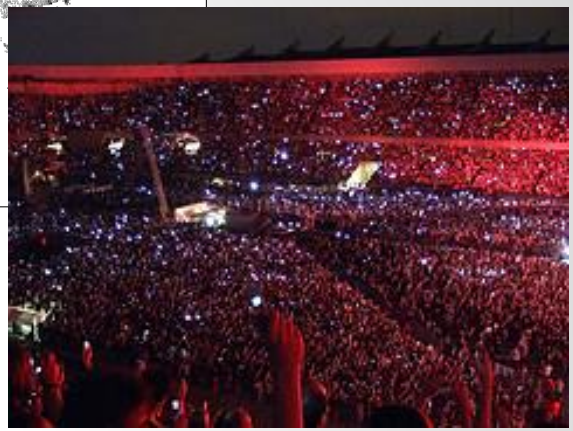
**Innovation!**

# From new requirements to network virtualization…



© Schiöberg et al. (ACM WebSci 2012)

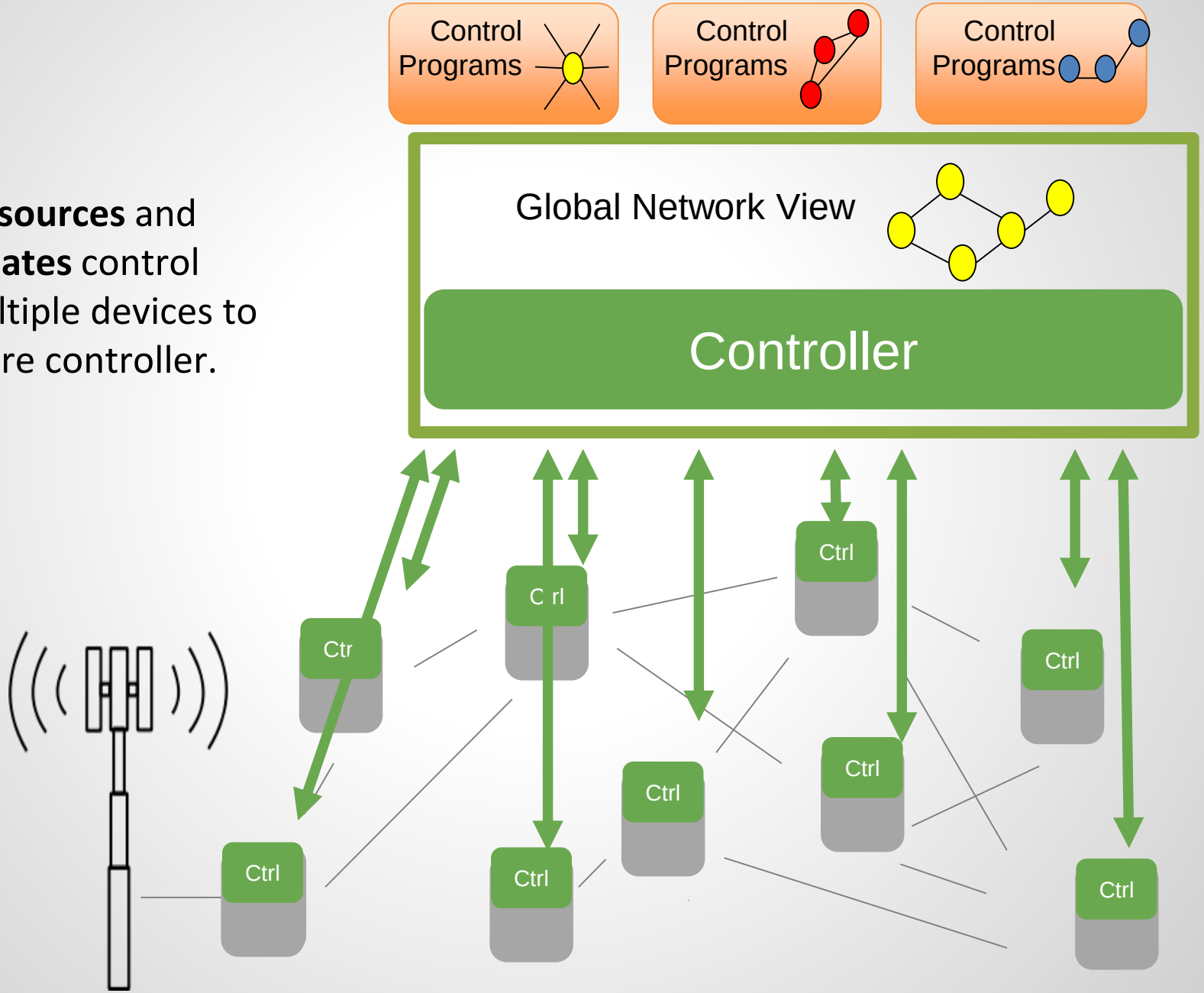


- Connectivity between fixed locations
- Simple applications like email and file transfer

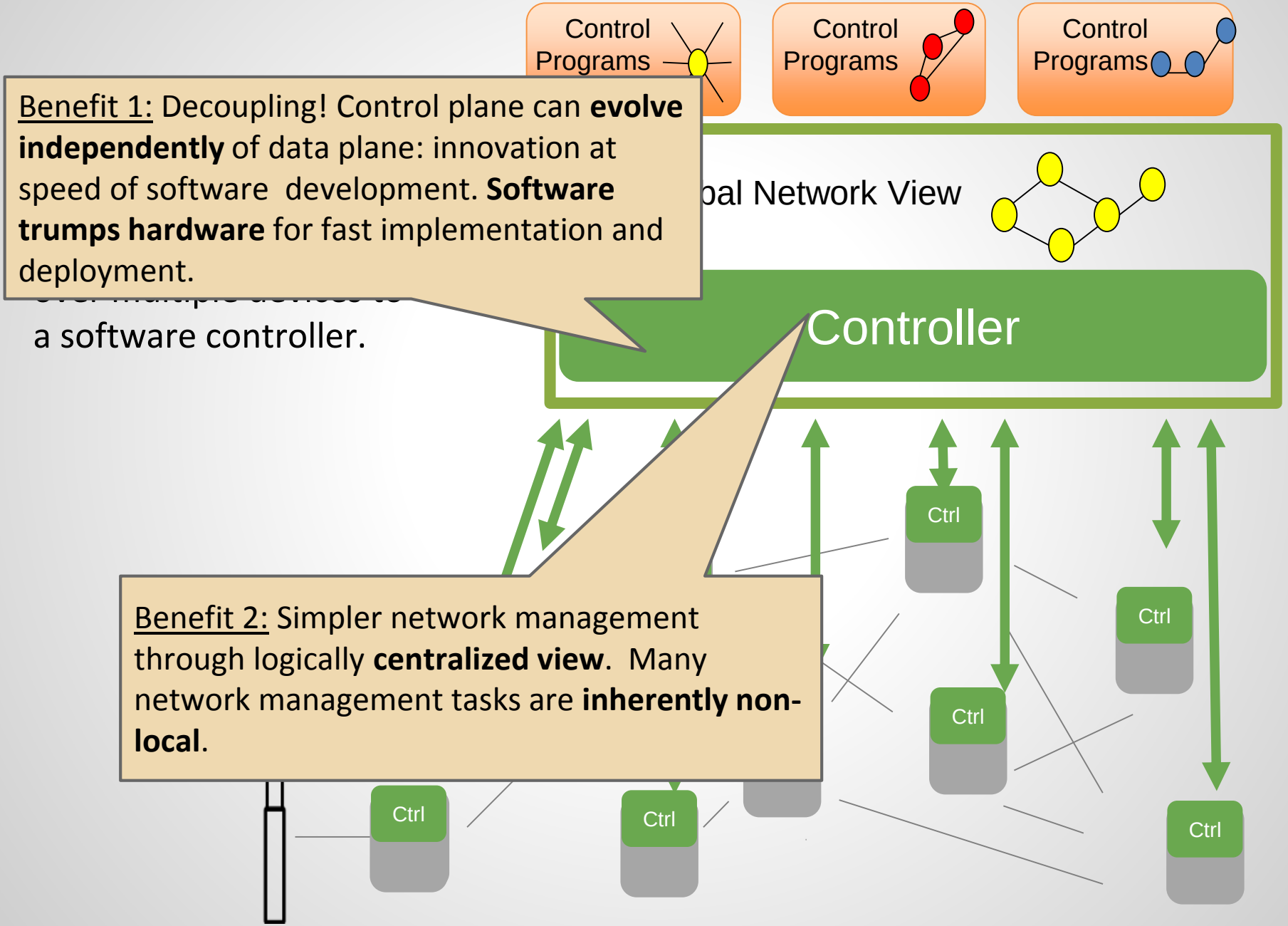
- Connectivity between humans, machines, datacenters, or *even things*
- Heterogeneous: e-commerce, VoD, science, etc.
- Wireless and mobile endpoints
- Mission-critical infrastructure

# Opportunity 2: Software-Defined Networks (SDNs)

SDN **outsources** and **consolidates** control over multiple devices to a software controller.

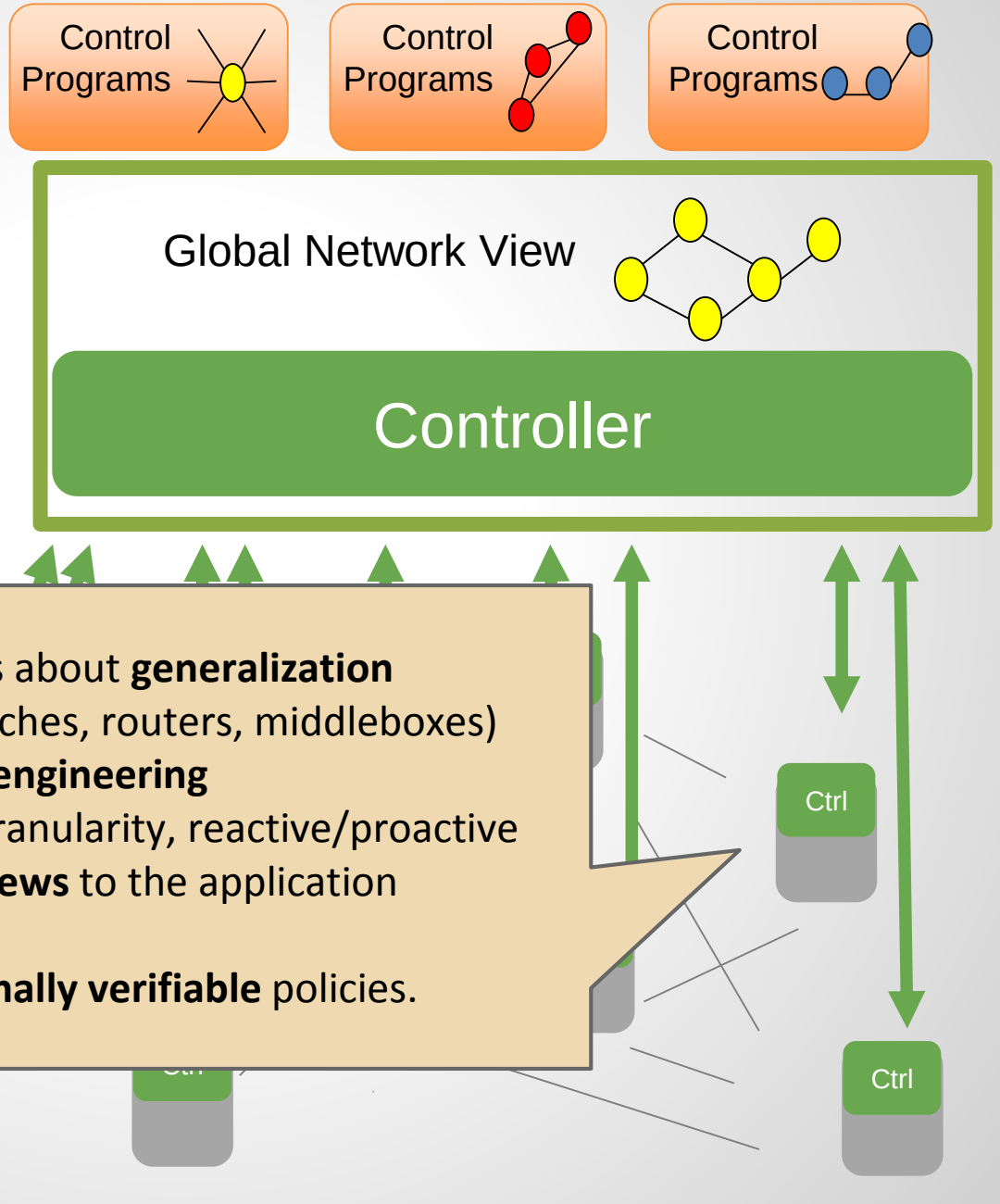


# Opportunity 2: Software-Defined Networks (SDNs)



# Opportunity 2: Software-Defined Networks (SDNs)

SDN **outsources** and **consolidates** control over multiple devices to a software controller.

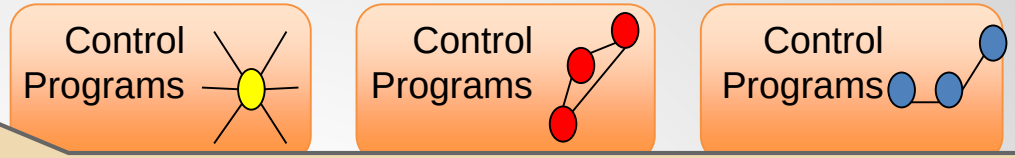



Benefit 3: Standard API OpenFlow is about **generalization**

- Generalize **devices** (L2-L4: switches, routers, middleboxes)
- Generalize **routing and traffic engineering**
- Generalize **flow-installation**: granularity, reactive/proactive
- General and logical **network views** to the application

Also: **match-action paradigm** = **formally verifiable** policies.

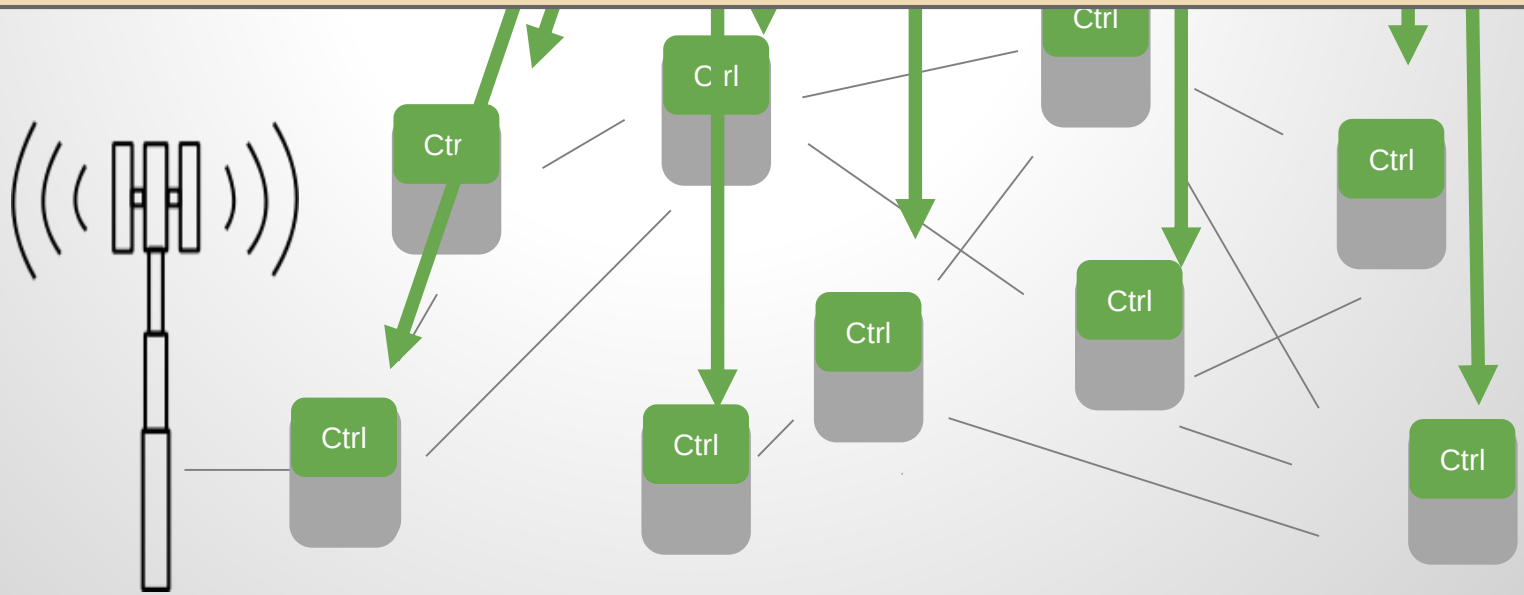
# Opportunity 2: Software-Defined Networks (SDNs)




SD
**co**
ov
a s

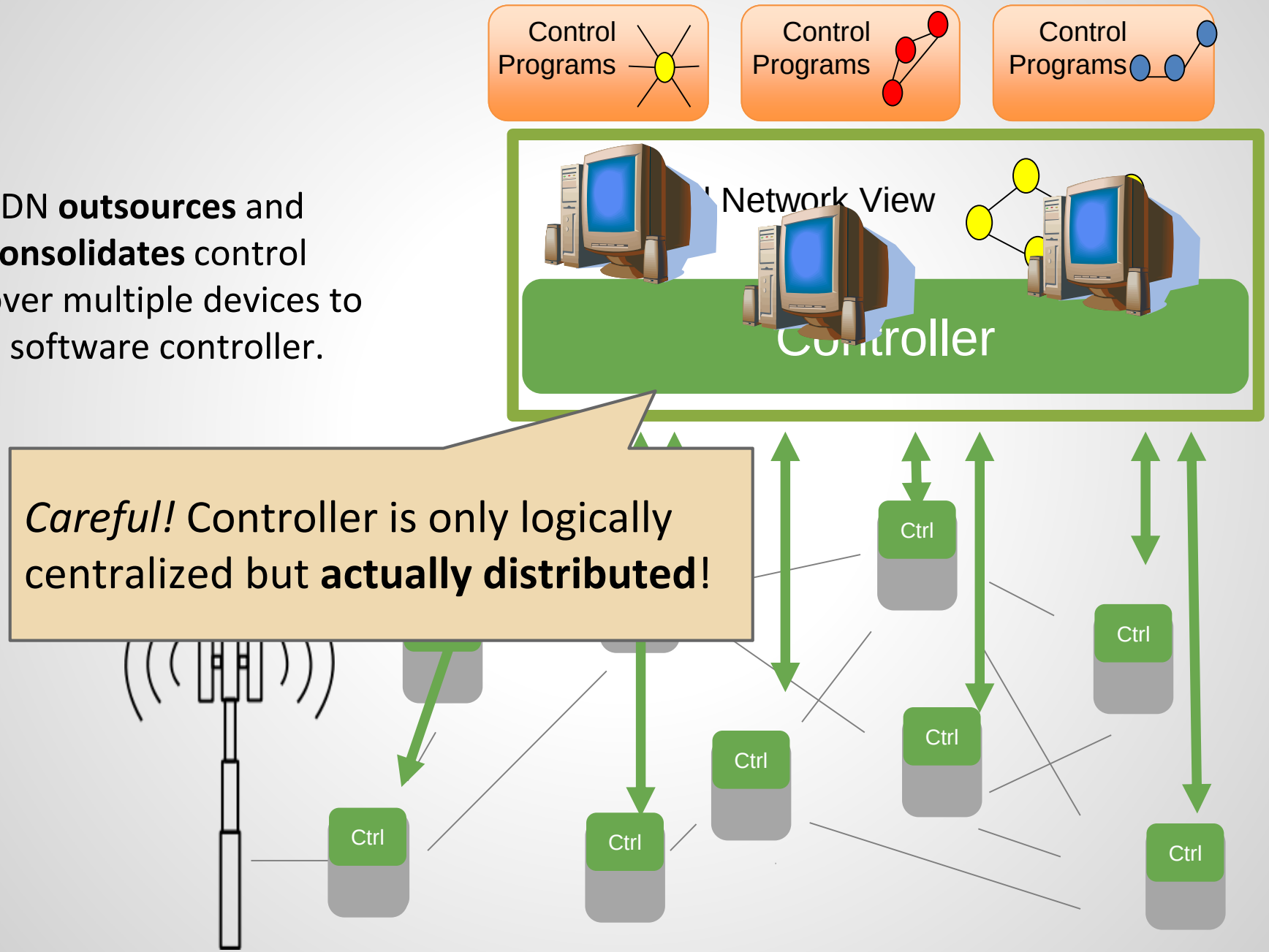
**But also challenges:**

- ❑ How to operate networks more adaptively without shooting in your foot?
- ❑ How to build a robust / scalable control plane?
- ❑ How to deploy this technology?
- ❑ Security: BSI project



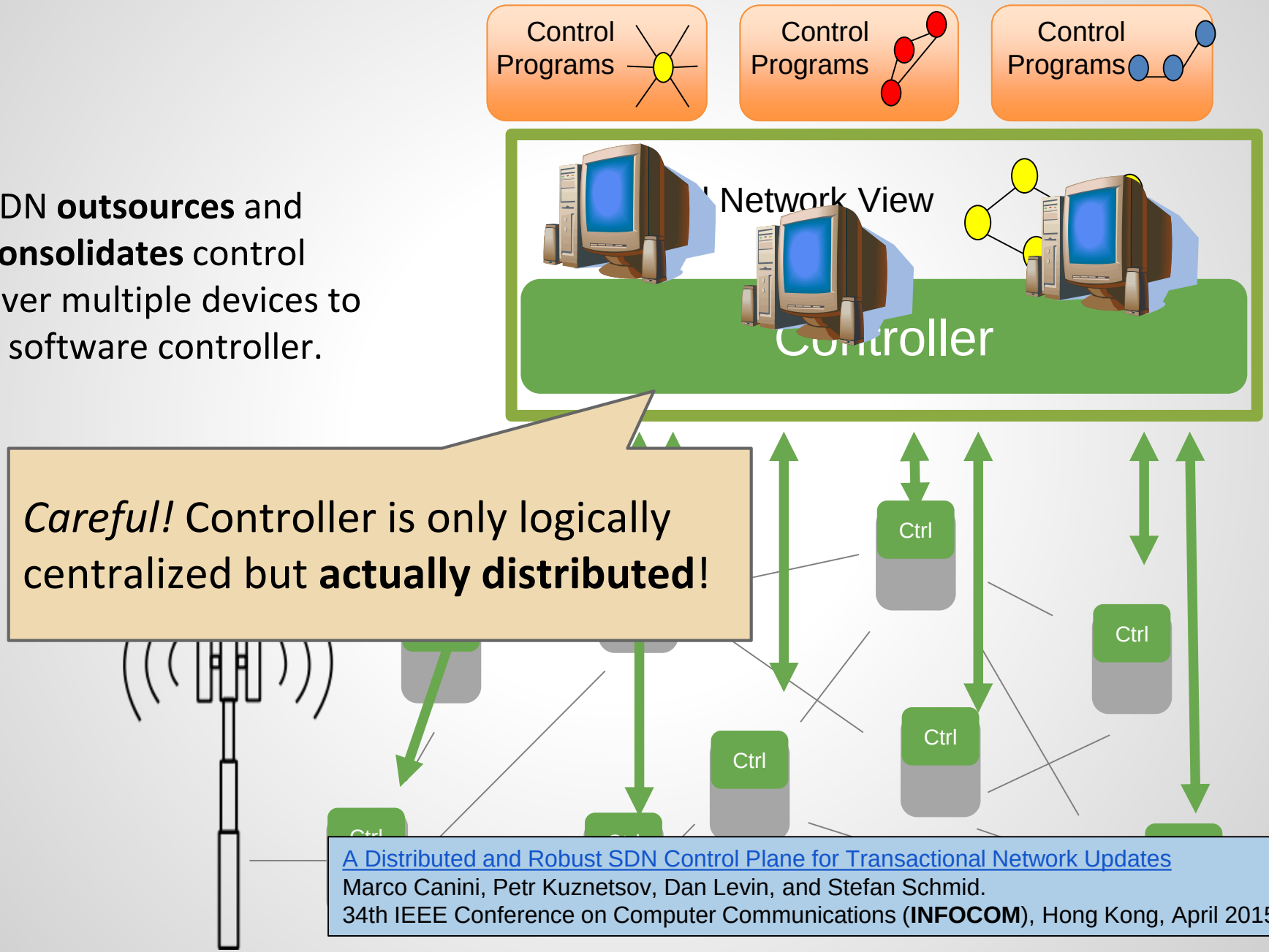
# Opportunity 2: Software-Defined Networks (SDNs)

SDN **outsources** and **consolidates** control over multiple devices to a software controller.



# Opportunity 2: Software-Defined Networks (SDNs)

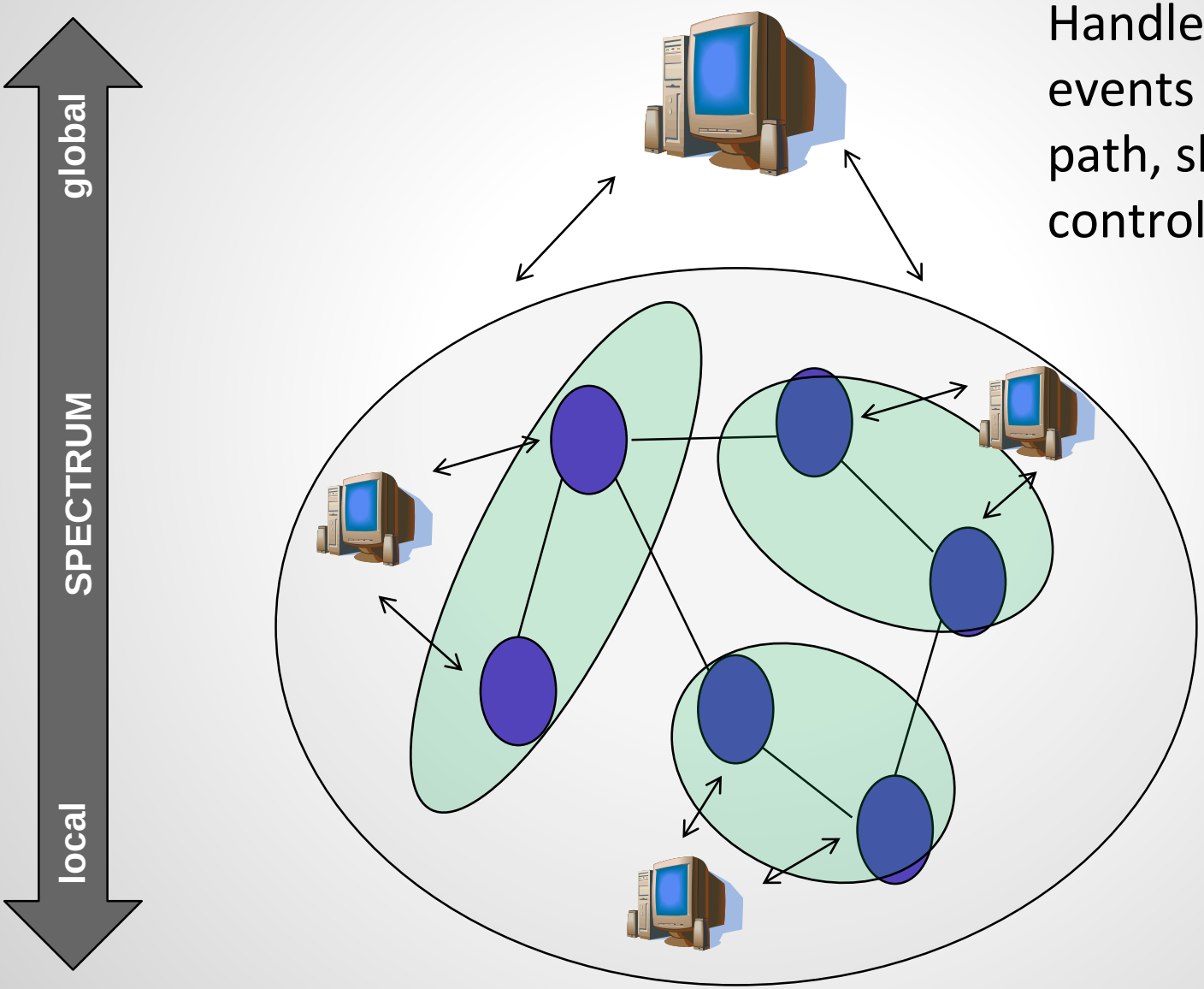
SDN **outsources** and **consolidates** control over multiple devices to a software controller.




*Careful!* Controller is only logically centralized but **actually distributed**!

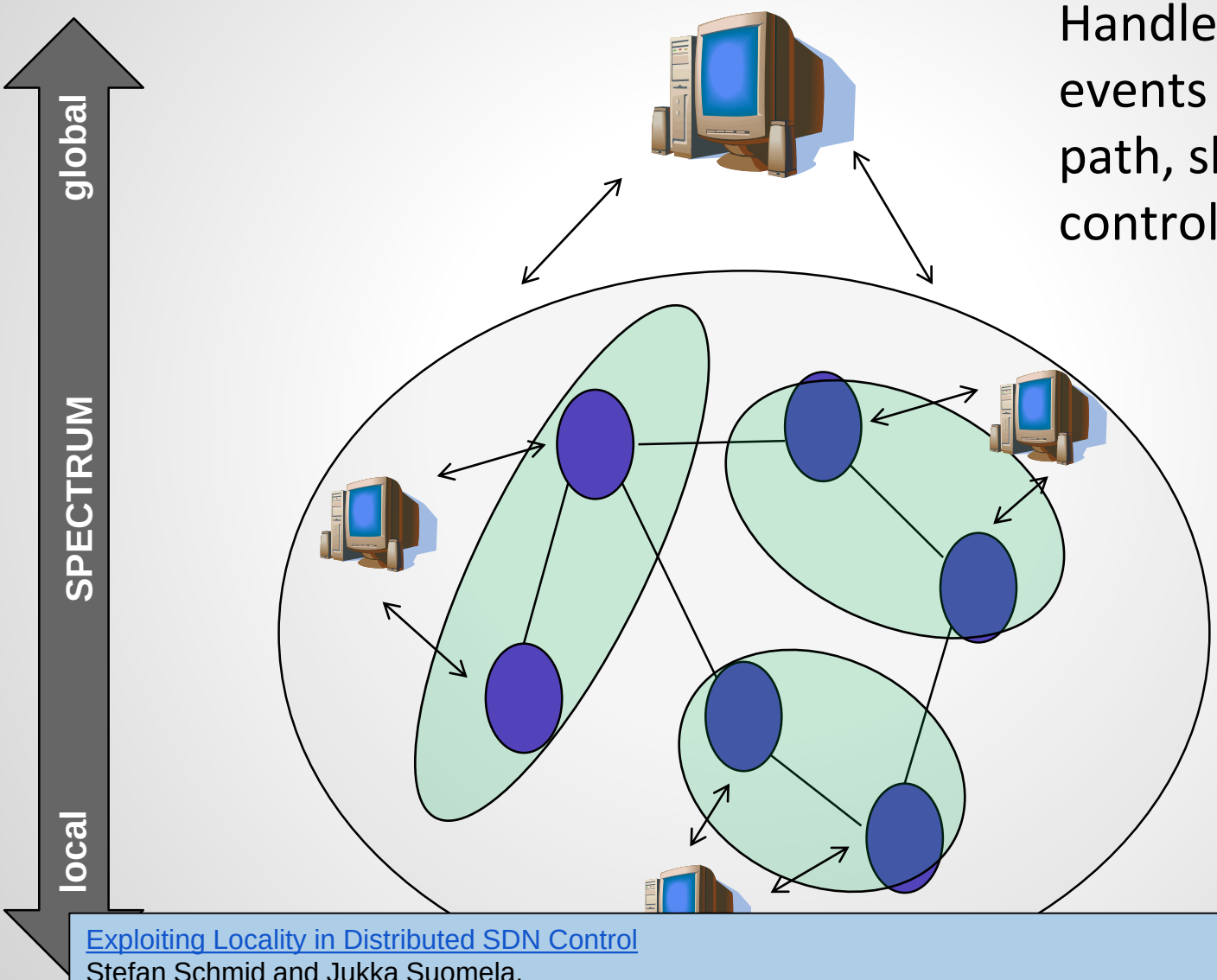
[A Distributed and Robust SDN Control Plane for Transactional Network Updates]
Marco Canini, Petr Kuznetsov, Dan Levin, and Stefan Schmid.
34th IEEE Conference on Computer Communications (**INFOCOM**), Hong Kong, April 2015..

# Locality-Aware and Fine-Grained Control




Handle frequent events close to data path, shield global controllers!

# Locality-Aware and Fine-Grained Control




Handle frequent events close to data path, shield global controllers!

Exploiting Locality in Distributed SDN Control
Stefan Schmid and Jukka Suomela.
ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (**HotSDN**), Hong Kong, China, August 2013.

# Challenges of More Flexible Networked Systems

1. Kraken: Predictable cloud application performance through adaptive virtual clusters

2. C3: Low tail latency in cloud data stores through replica selection

3. Peacock: Consistent network updates

4. Panopticon: How to introduce these innovative technologies in the first place? Case study: SDN

# Challenges of More Flexible Networked Systems

1. Kraken: Predictable cloud applicati…rformance through adaptive virtual clust…

SIGCOMM CCR 2015
INFOCOM 2016

2. C3: Low tai…tency in cloud data stores through replica selection

USENIX NSDI 2015

3. Peacock: Co…twork updates

ACM SIGMETRICS 2016
IEEE/IFIP DSN 2016
ACM PODC 2015

4. Panopticon: How to int…duce these innovative technologies in t…irst place? Case study: SDN

USENIX ATC 2014

# Challenges of More Flexible Networked Systems

1. **Kraken: Predictable cloud application performance through adaptive virtual clusters**

2. C3: Low tail latency in cloud data stores through replica selection

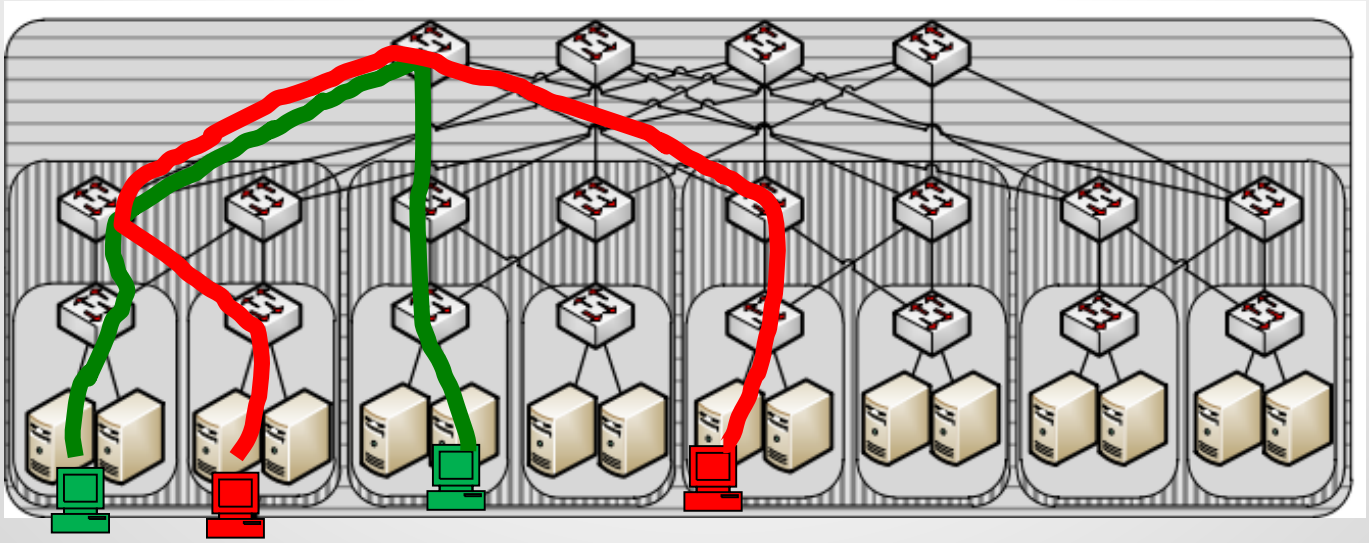
3. Peacock: Consistent network updates

4. Panopticon: How to introduce these innovative technologies in the first place? Case study: SDN

# Cloud Computing + Networking?!
# *Network matters!*

- Example: Batch Processing Applications such as Hadoop
  - **Communication intensive**: e.g., shuffle phase
  - Example Facebook: 33% of **execution time** due to communication

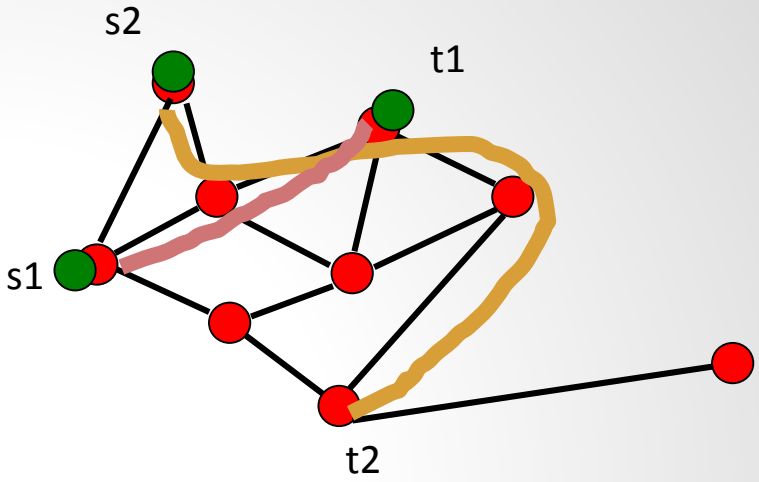
- For predictable preformance in shared cloud: need explicit bandwidth reservations: an **embedding problem**!



# Let's Exploit Allocation Flexibilities to Maximize Utilization!

Start simple: exploit flexible routing between given VMs

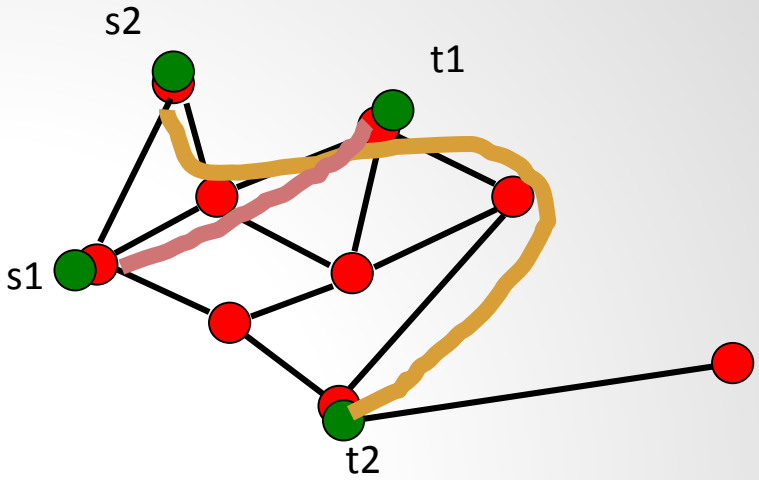
- Integer multi-commodity flow problem with 2 flows?



# Let's Exploit Allocation Flexibilities to Maximize Utilization!

Start simple: exploit flexible routing between given VMs

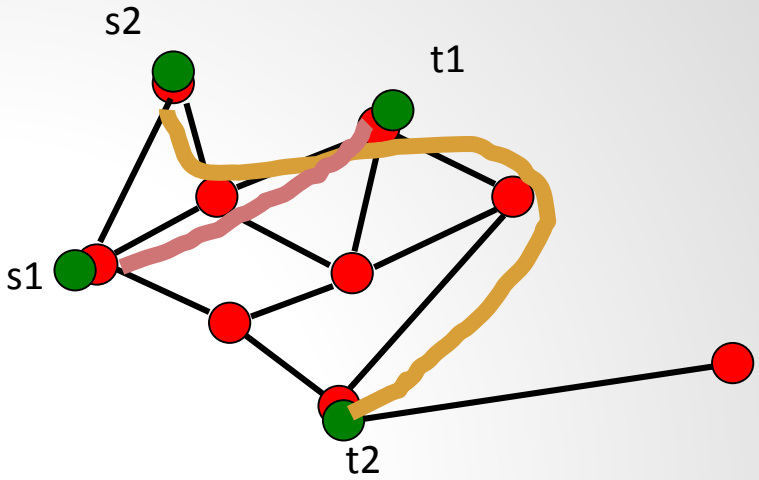
- Integer multi-commodity flow problem with 2 flows?
- Oops: NP-hard

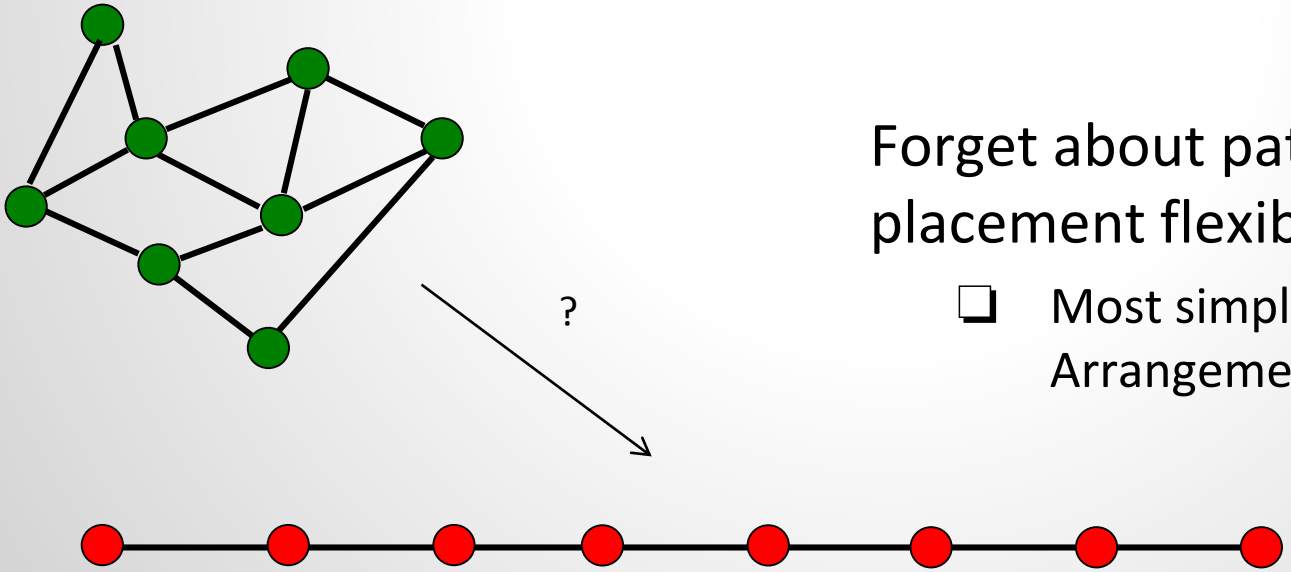


# Let's Exploit Allocation Flexibilities to Maximize Utilization!

Start simple: exploit flexible routing between given VMs

- Integer multi-commodity flow problem with 2 flows?
- Oops: NP-hard



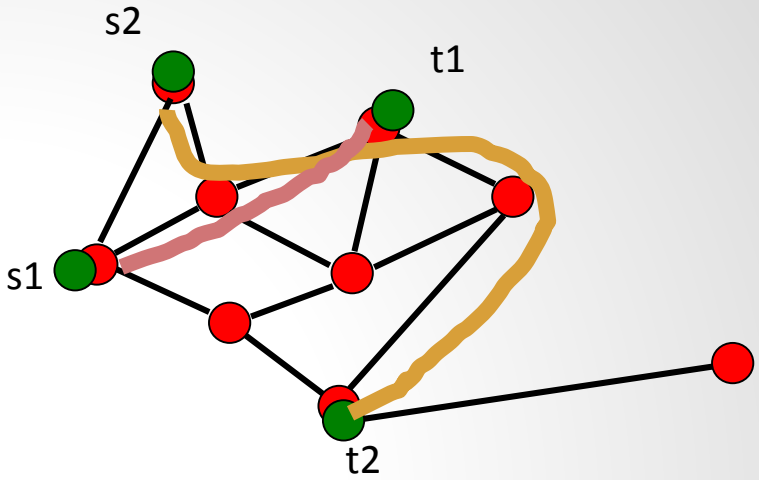

Forget about paths: exploit VM placement flexibilities!

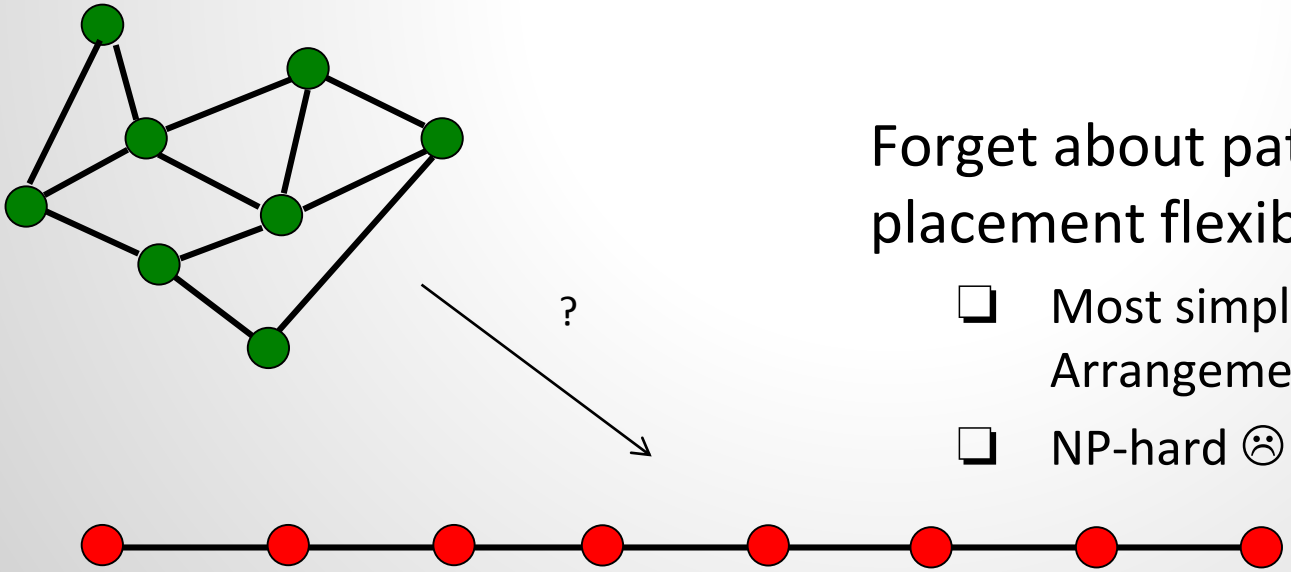
- Most simple: Minimum Linear Arrangement without capacities

# Let's Exploit Allocation Flexibilities to Maximize Utilization!

Start simple: exploit flexible routing between given VMs

- ❑ Integer multi-commodity flow problem with 2 flows?
- ❑ Oops: NP-hard



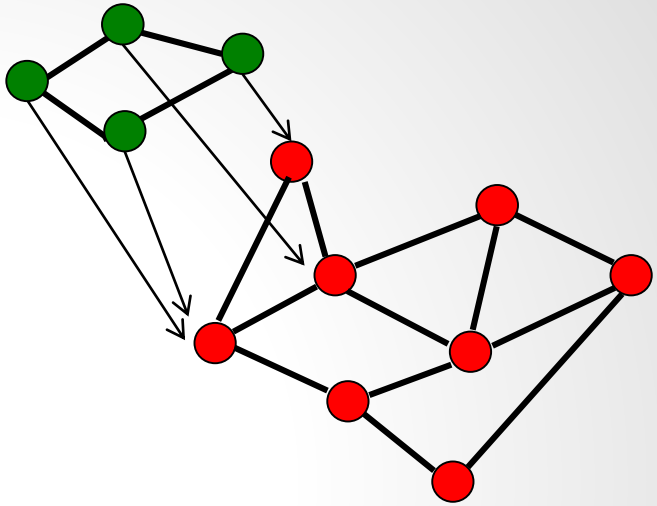

Forget about paths: exploit VM placement flexibilities!

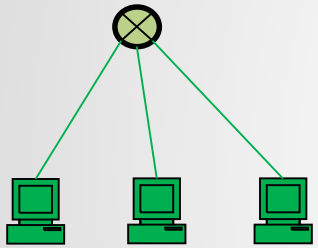
- ❑ Most simple: Minimum Linear Arrangement without capacities
- ❑ NP-hard ☹

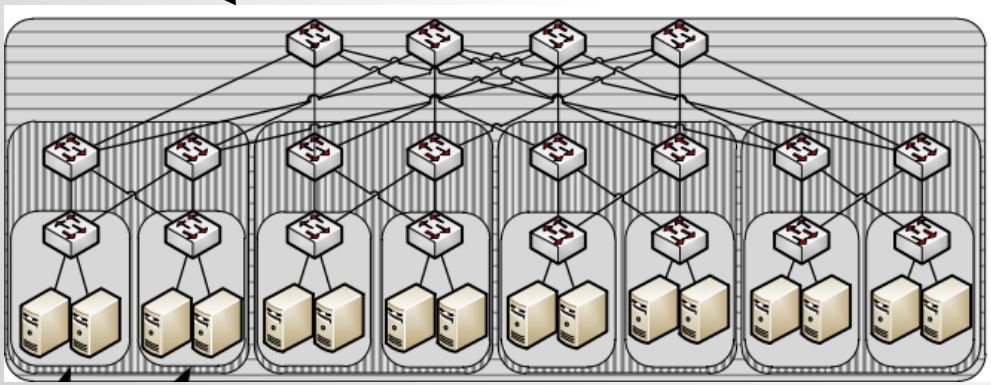
# Theory vs Practice

**Goal in theory:**

Embed as general as possible ***guest graph*** to as general as possible ***host graph***





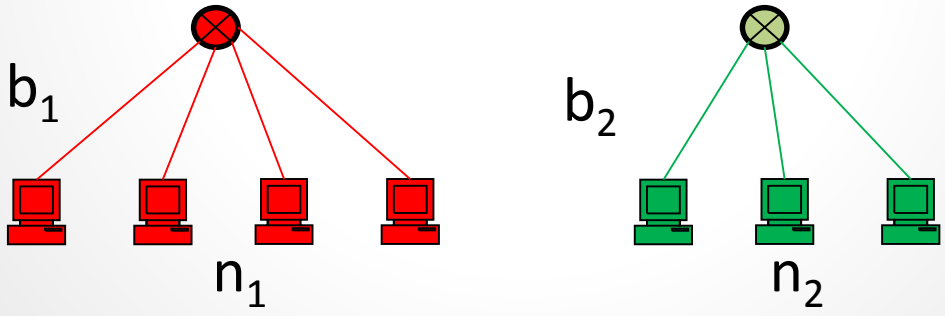


**Reality:**

Datacenters, WANs, etc. exhibit much **structure** that can be exploited! But also guest networks come with **simple specifications**

# Virtual Clusters

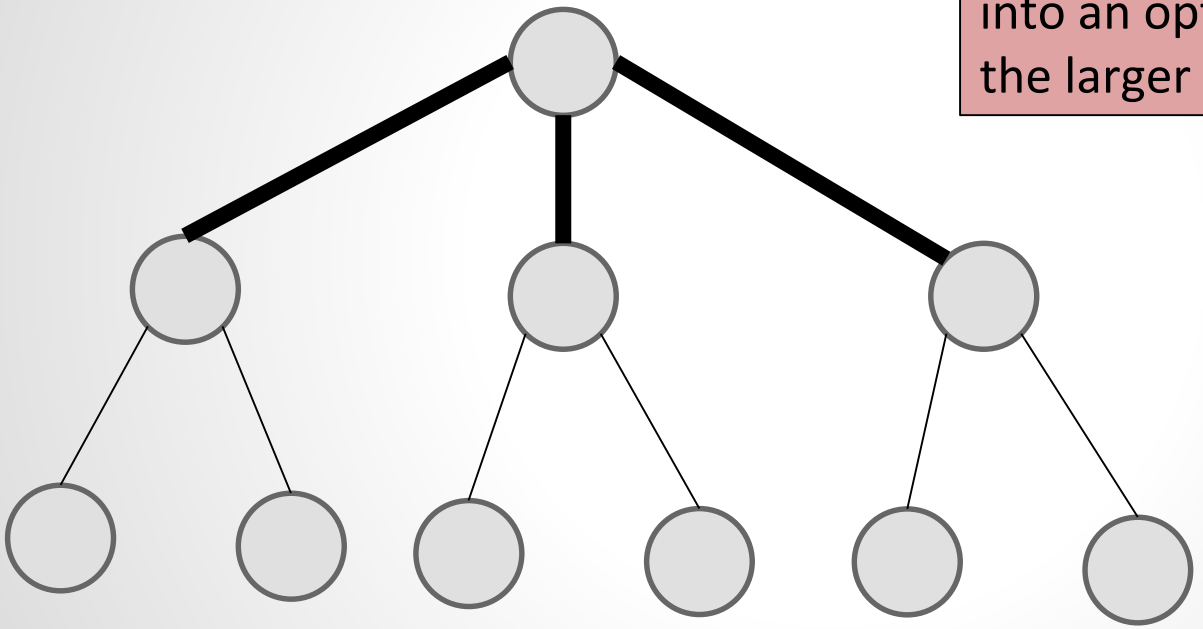
- A prominent abstraction for batch-processing applications: Virtual Cluster *VC(n,b)*
  - Connects *n* virtual machines to a «logical» switch with bandwidth guarantees *b*
  - A simple abstraction



# How to embed a Virtual Cluster in a Fat-Tree?

- Example: dynamic programming

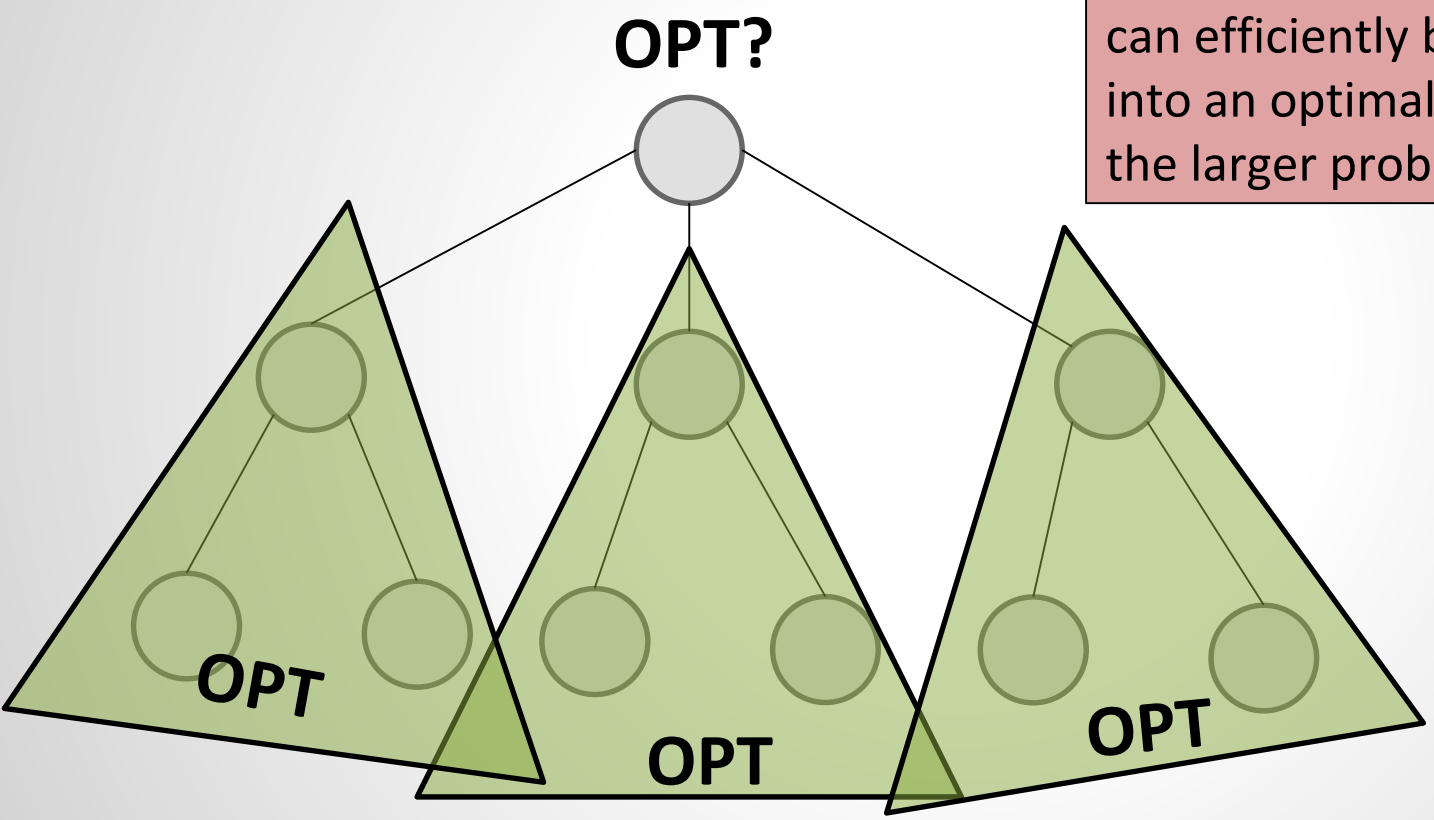
Dynamic Program = optimal solutions for subproblems can efficiently be combined into an optimal solution for the larger problem!



# How to embed a Virtual Cluster in a Fat-Tree?

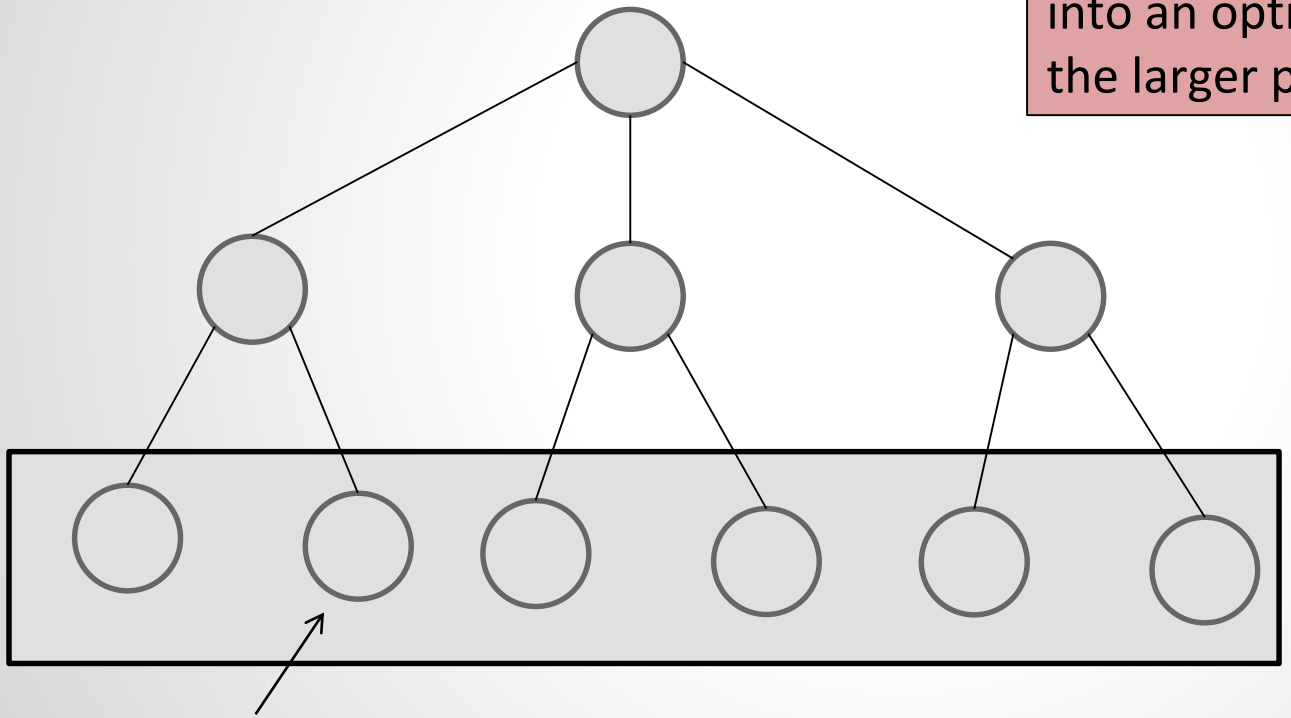
- Example: dynamic programming

Dynamic Program = optimal solutions for subproblems can efficiently be combined into an optimal solution for the larger problem!



# How to embed a Virtual Cluster in a Fat-Tree?

Dynamic Program = optimal solutions for subproblems can efficiently be combined into an optimal solution for the larger problem!



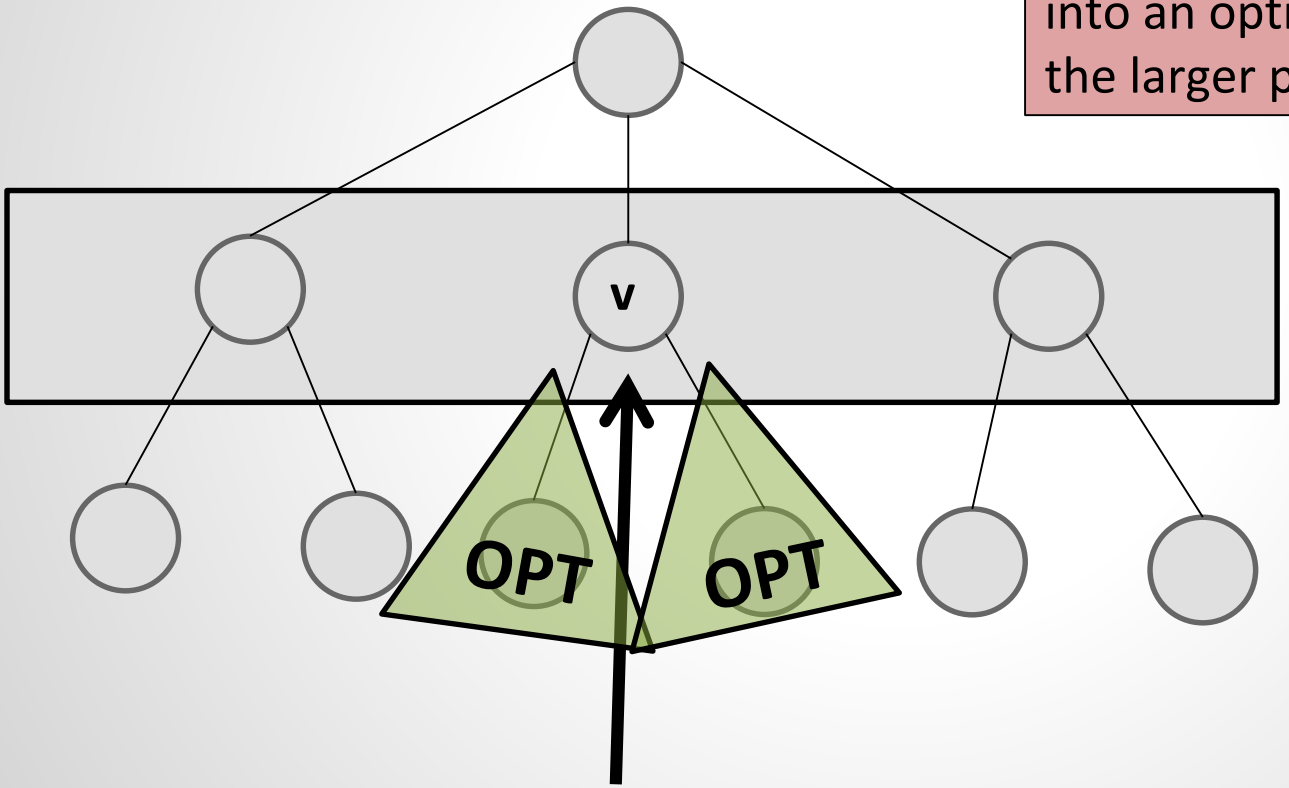
**t = 0: solve leaves!**

How to optimally embed x VMs here, $x \in \{0, ..., n\}$?

Cost = 0 or $\infty$!

# How to embed a Virtual Cluster in a Fat-Tree?

Dynamic Program = optimal solutions for subproblems can efficiently be combined into an optimal solution for the larger problem!



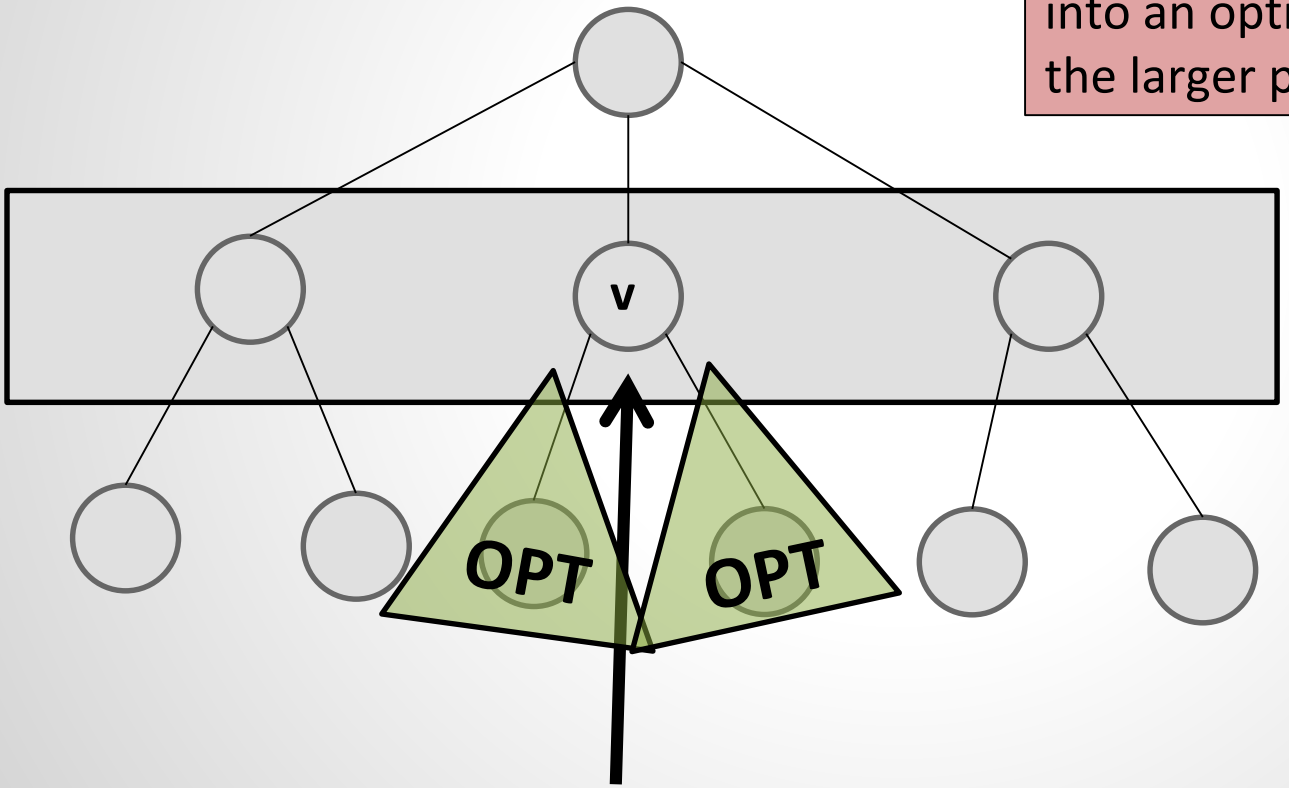

**t = 1: solve height 1!**

$Cost[x] = \min_y Cost[y] + Cost[x-y]$

\+ cross-traffic + connections to v

# How to embed a Virtual Cluster in a Fat-Tree?

Dynamic Program = optimal solutions for subproblems can efficiently be combined into an optimal solution for the larger problem!




**t = 1: solve height 1!**

Cost[x] = $\min_y$ Cost[y] + Cost[x-y]

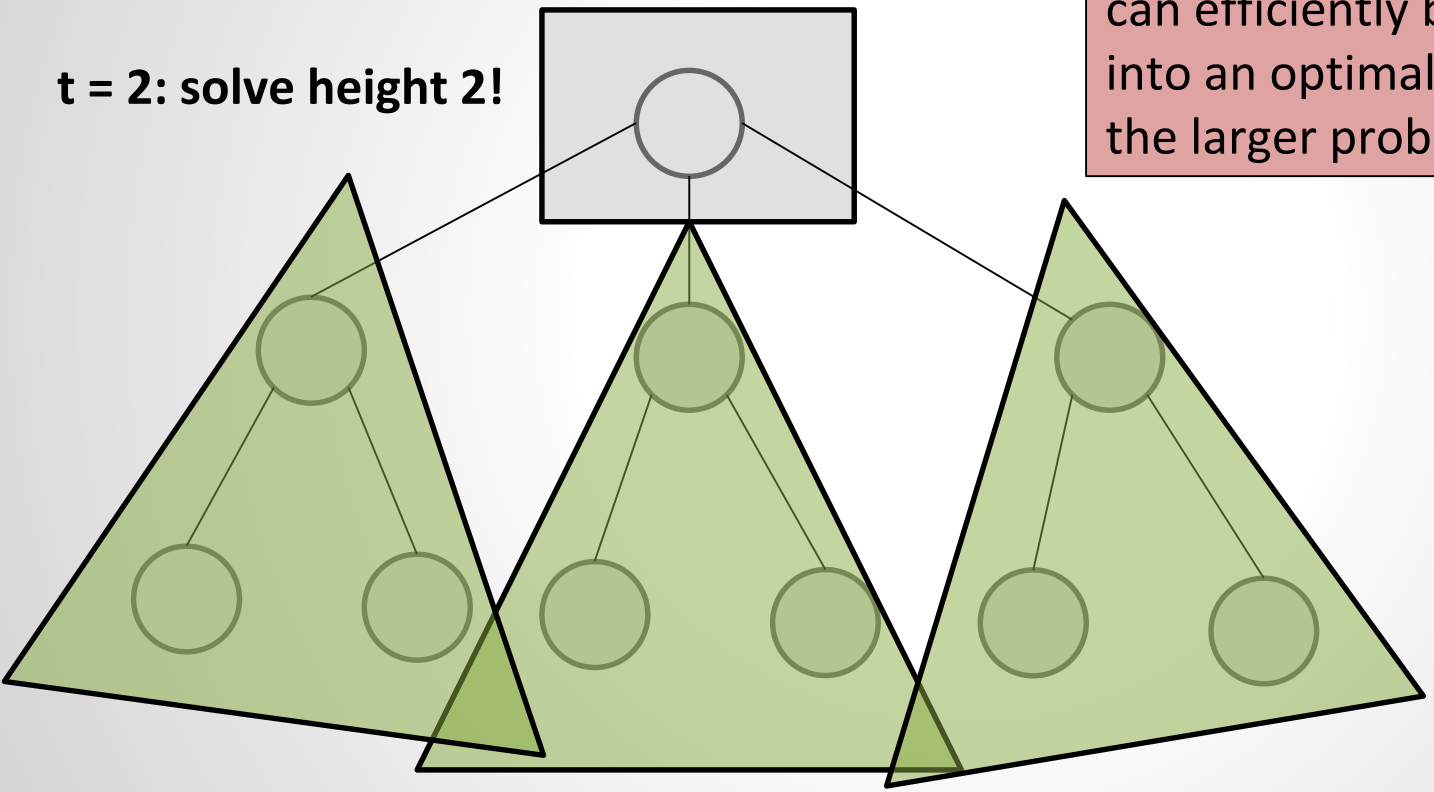
+ cross-traffic + connections to v

**Or just account on upward link (number of leaving links!)**

# How to embed a Virtual Cluster in a Fat-Tree?

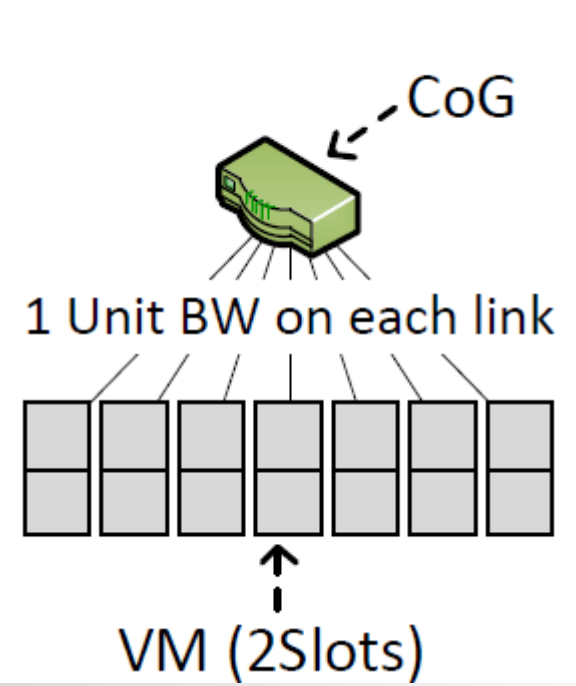
Dynamic Program = optimal solutions for subproblems can efficiently be combined into an optimal solution for the larger problem!

**t = 2: solve height 2!**

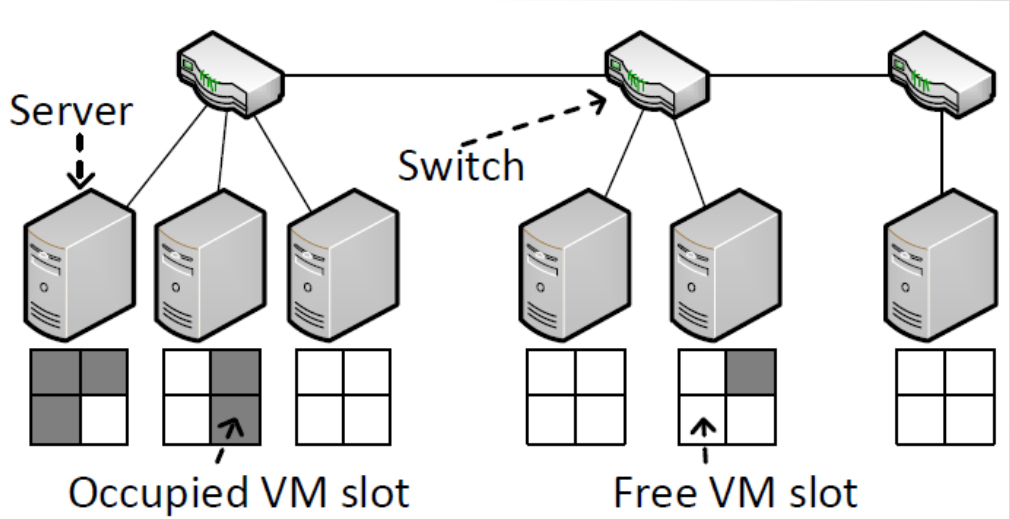


# How to embed a Virtual Cluster in a General Graph?

How to embed?




Guest Graph

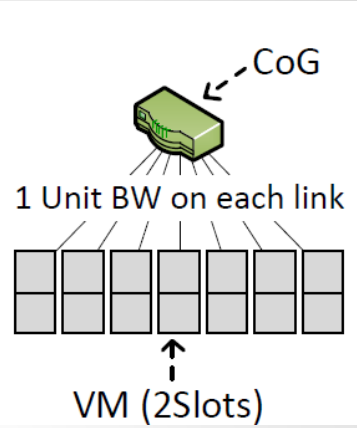


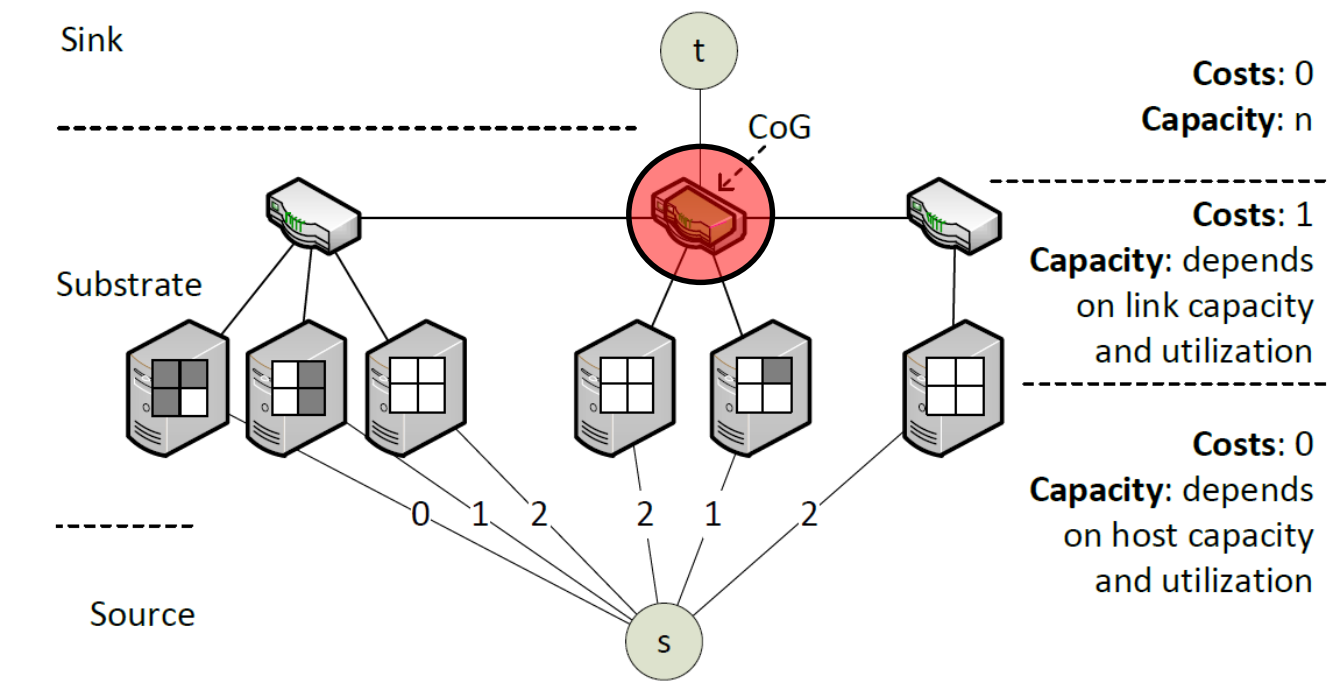

Host Graph

# How to embed a Virtual Cluster in a General Graph?

## Algorithm:

- Try all possible locations for virtual switch
- Extend network with artificial source s and sink t
- Add capacities
- Compute min-cost max-flow from s to t
  (or simply: min-cost flow of volume n)

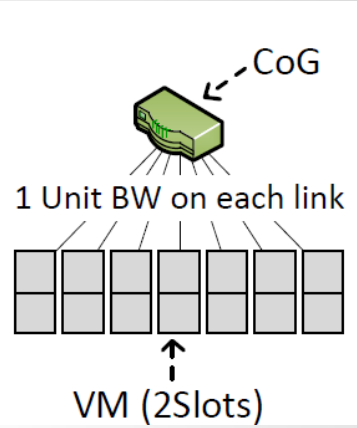


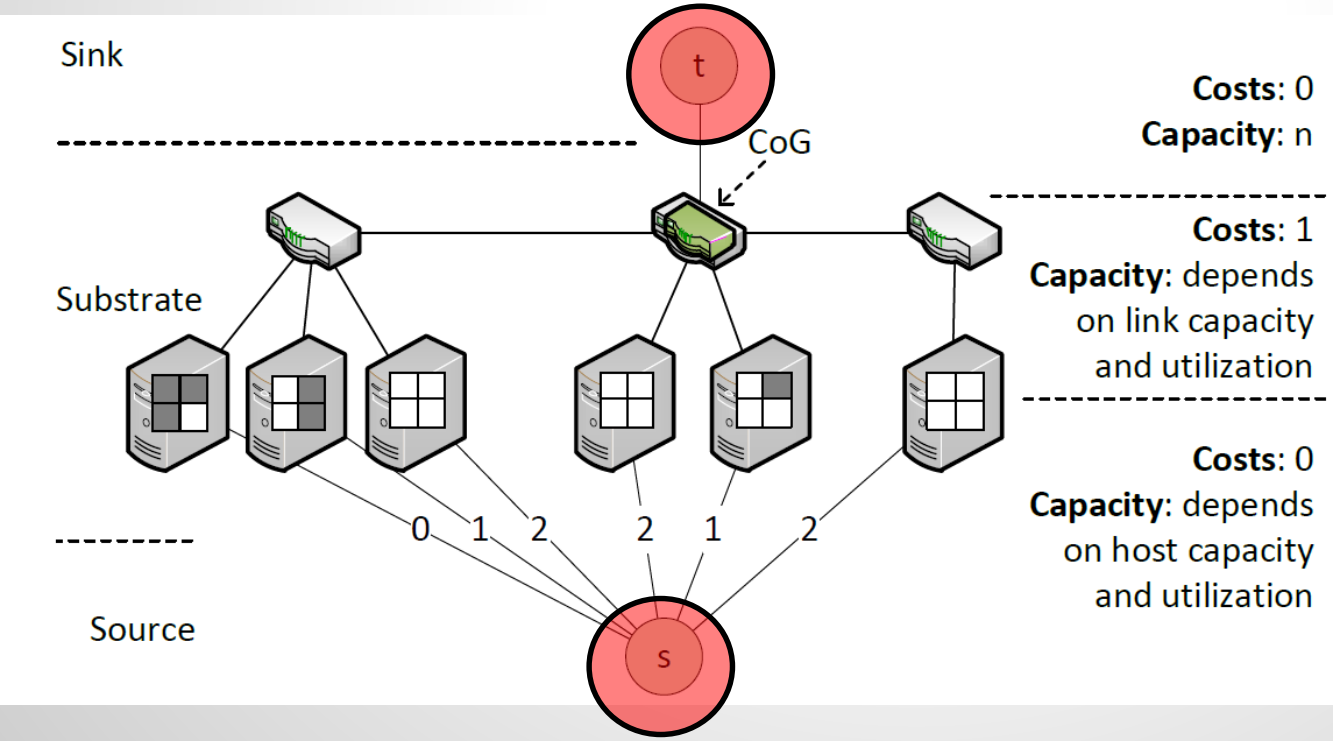



# How to embed a Virtual Cluster in a General Graph?

## Algorithm:

- Try all possible locations for virtual switch
- Extend network with artificial source s and sink t
- Add capacities
- Compute min-cost max-flow from s to t
  (or simply: min-cost flow of volume n)

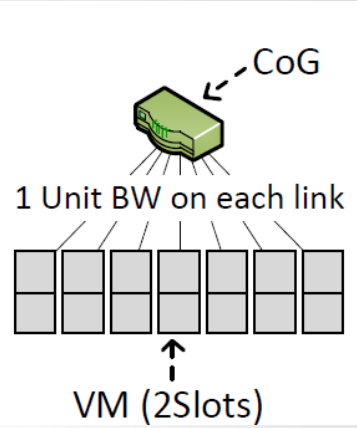


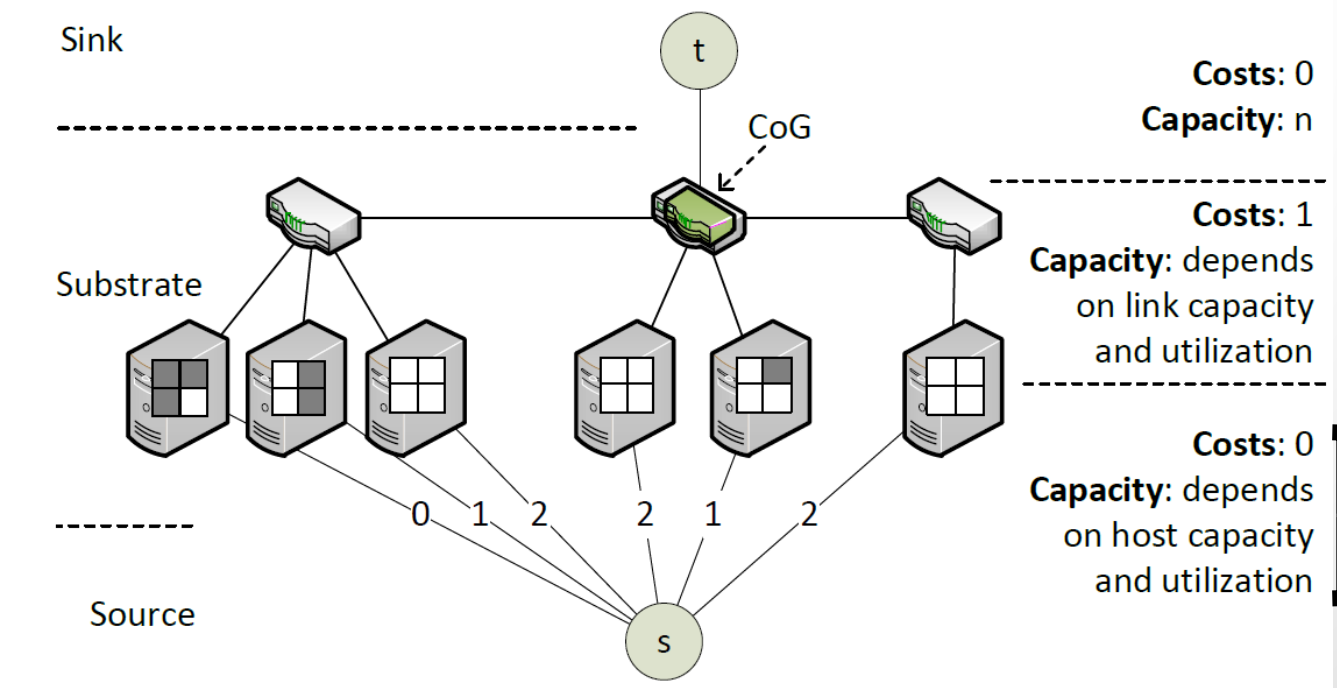



# How to embed a Virtual Cluster in a General Graph?

## Algorithm:

- Try all possible locations for virtual switch
- Extend network with artificial source s and sink t
- Add capacities
- Compute min-cost max-flow from s to t
  (or simply: min-cost flow of volume n)



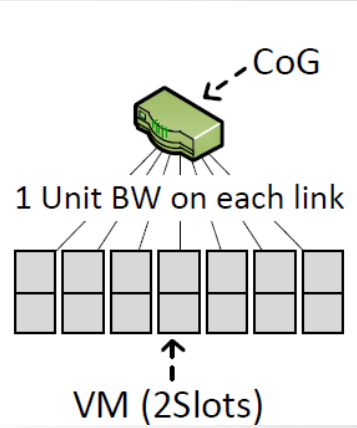

**enough to embed n VMs**

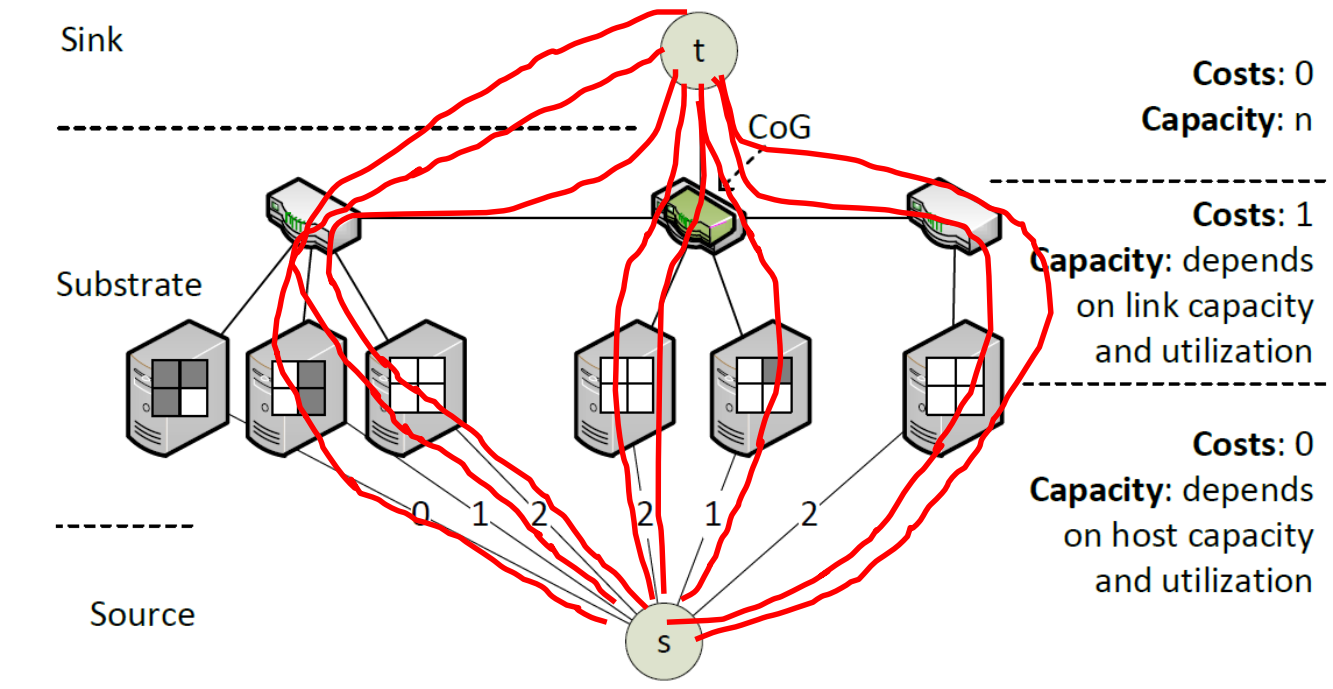
**capacity = floor(available resources / unit demand)**

# How to embed a Virtual Cluster in a General Graph?

## Algorithm:

- Try all possible locations for virtual switch
- Extend network with artificial source s and sink t
- Add capacities
- Compute min-cost max-flow from s to t
  (or simply: min-cost flow of volume n)



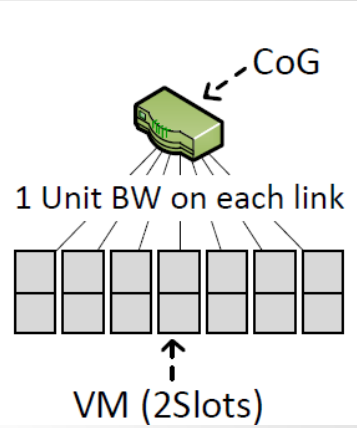



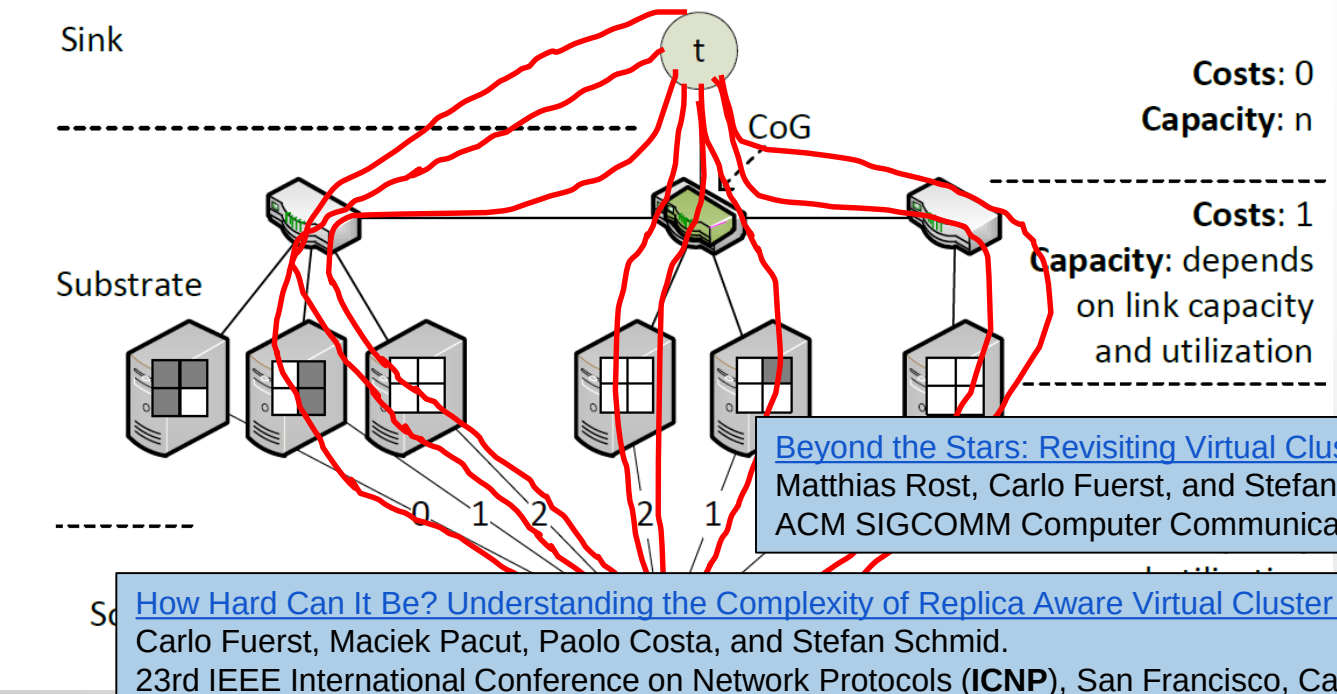

**Guaranteed integer if links are integer! (E.g., successive shortest paths)**

# How to embed a Virtual Cluster in a General Graph?

## Algorithm:

- Try all possible locations for virtual switch
- Extend network with artificial source s and sink t
- Add capacities
- Compute min-cost max-flow from s to t
  (or simply: min-cost flow of volume n)




Guaranteed integer


Beyond the Stars: Revisiting Virtual Cluster Embeddings
Matthias Rost, Carlo Fuerst, and Stefan Schmid.
ACM SIGCOMM Computer Communication Review (**CCR**), July 2015..

How Hard Can It Be? Understanding the Complexity of Replica Aware Virtual Cluster Embeddings
Carlo Fuerst, Maciek Pacut, Paolo Costa, and Stefan Schmid.
23rd IEEE International Conference on Network Protocols (**ICNP**), San Francisco, California, USA, November 2015.

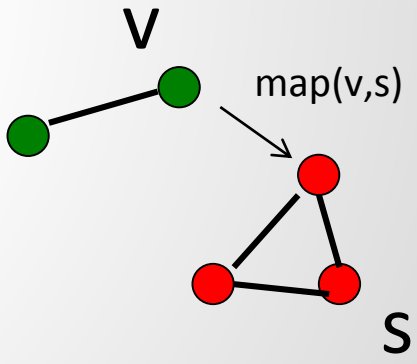
# Rigorous Solutions for the General Embedding Problem: MIP

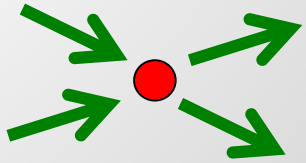
Recipe for VNEP formulation :

- ❑ A (linear) objective function (e.g., load or footprint)
- ❑ A set of (linear) constraints
- ❑ Feed it to your favorite solver (CPLEX, Gurobi, etc.)

Details:

- ❑ Introduce binary variables ***map(v,s)*** to map virtual nodes *v* on substrate node *s*
- ❑ Introduce flow variables for paths (splittable or not?)
- ❑ Ensure **flow conservation**: all flow entering a node must leave the node, unless it is the source or the destination



# Rigorous Solutions for the General Embedding Problem: MIP

**Constants:**

Substrate Vertices : $V_s$
Substrate Edges : $E_s : V_s \times V_s$
Unique : $uni\_check_s : \forall (s_1, s_2) \in E_s : (s_2, s_1) \notin E_s$
SNode Capacity : $snc(s) \rightarrow \mathbb{R}^+, s \in V_s$
SLink Capacity : $slc(e_s) \rightarrow \mathbb{R}^+, e_s \in E_S$

Requests : $R$
Virtual Vertices : $V_v(r), r \in R$
Virtual Edges : $E_v(r) :\rightarrow V_v(r) \times V_v(r), r \in R$
Unique : $uni\_check_v : \forall r \in R, (v_1, v_2) \in E_v(r) : (v_2, v_1) \notin E_v(r)$
VNode Demand : $vnd(r, v) \rightarrow \mathbb{R}^+, r \in R, v \in V_v(r)$
VEdge Demand : $vld(r, e_v) \rightarrow \mathbb{R}^+, r \in R, e_v \in E_v(r)$

Edges-Reverse : $ER_s : \forall (s_1, s_2) \in E_s \exists (s_2, s_1) \in ER_s \wedge |E_s| = |ER_s|$
Edges-Bidirectional : $EB_s : E_s \cup ER_s$
Migration Cost : $mig\_cost(r, v, s) \rightarrow \mathbb{R}^{+ \ |V_v(r)| \times |V_s|}, r \in R, v \in V_v(r), s \in V_s$
Possible Placements : $place(r, v, s) \rightarrow \{0, 1\}^{|V_v(r)| \times |V_s|}, r \in R, v \in V_v(r), s \in V_s$

**Variables:**

Node Mapping : $n\_map(r, v, s) \in \{0, 1\}, r \in R, v \in V_v(r), s \in V_s$
Flow Allocation : $f\_alloc(r, e, eb) \geq 0, r \in R, e \in E_v(r), eb \in EB_s$

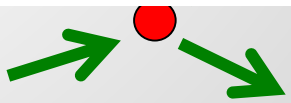
**Constraints:**

Each Node Mapped : $\forall r \in R, v \in V_v(r) : \sum_{s \in V_s} n\_map(r, v, s) \cdot place(r, v, s) = 1$
Feasible : $\forall s \in V_s : \sum_{r \in R, v \in V_v(r)} n\_map(r, v, s) \cdot vnd(r, v) \leq snc(s)$
Guarantee Link Realization : $\forall r \in R, (v_1, v_2) \in E_v(r), s \in V_s \sum_{(s,s_2) \in V_s \times V_s \cap EB_s} f\_alloc(r, v_1, v_2, s, s_2) -$
$\sum_{(s_1,s) \in V_s \times V_s \cap EB_s} f\_alloc(r, v_1, v_2, s_1, s) = vld(r, v_1, v_2) \cdot (n\_map(r, v_1, s) - n\_map(r, v_2, s))$
Realize Flows : $\forall (s_1, s_2) \in E_s \sum_{r \in R, (v_1, v_2)} f\_alloc(r, v_1, v_2, s_1, s_2) + f\_alloc(r, v_1, v_2, s_2, s_1) \leq slc(s_1, s_2)$

**Objective function:**

Minimize Embedding Cost : $min : \sum_{r \in R, (v_1, v_2) \in E_v(r), (s_1, s_2) \in E_s} f\_alloc(r, v_1, v_2, s_1, s_2) + f\_alloc(r, v_1, v_2, s_2, s_1)$

entering a node must leave the node,
unless it is the source or the destination

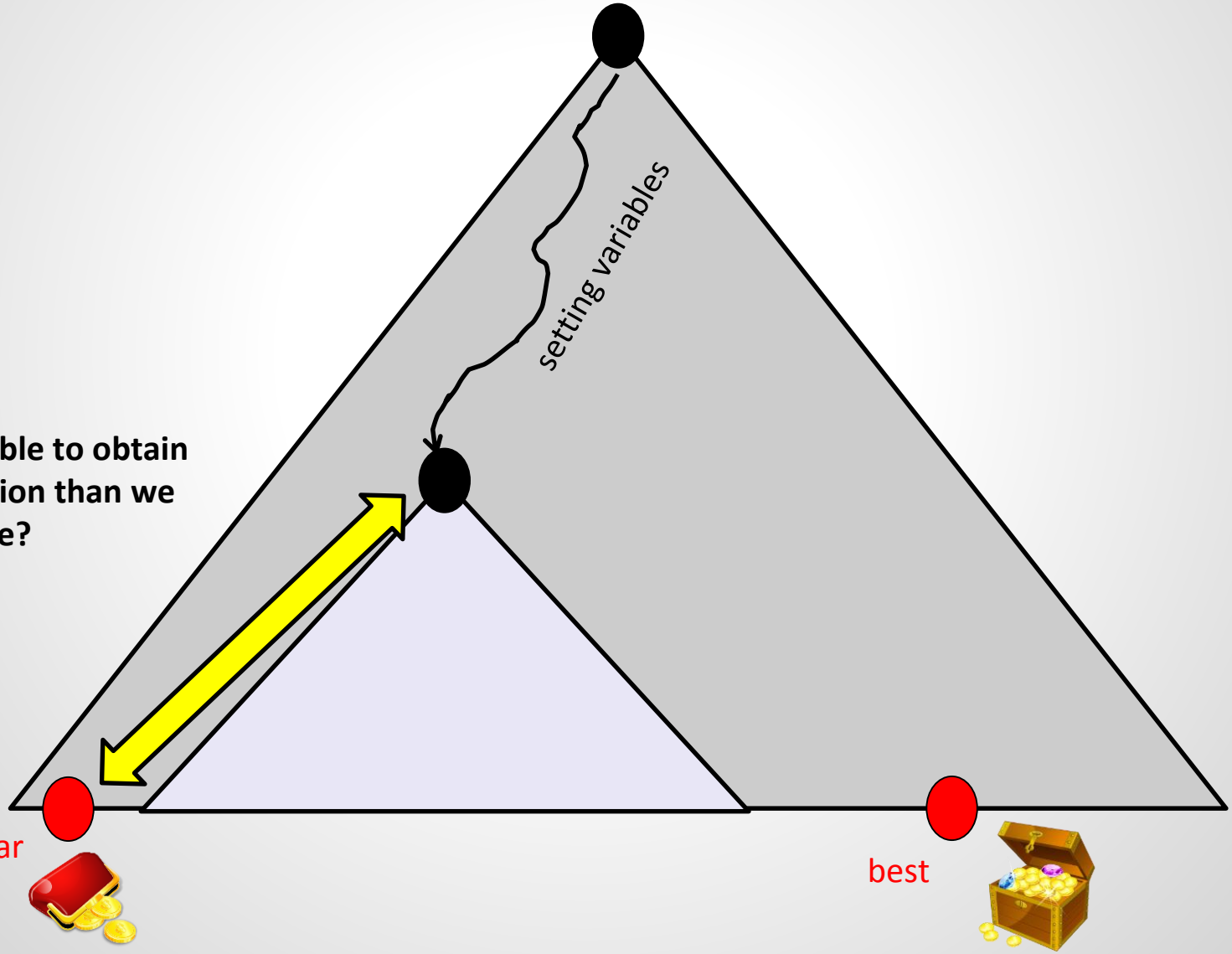


# Mixed Integer Programs (1)

- MIPs can be quite fast
  - For pure integer programs, SAT solvers likely faster
- However, that's not the end of the story: MIP ≠ MIP
  - The specific formulation matters!
- For example: many solvers use relaxations
  - Make integer variables continuous: resulting linear programs (LPs) can be solved in polynomial time!
  - How good can solution in this subtree (given fixed variables) be at most? (More flexibility: solution can only be better!)
  - If already this is worse than currently best solution, we can cut!
- Relaxations can also be used as a basis for heuristics
  - E.g., round fractional solutions to closest integer?

# Mixed Integer Programs (2)

Branch & bound tree:



# Mixed Integer Programs (3)

- Recall: Relaxations useful if they give good bounds
- However it's hard to formulate a MIP for VNEP which yields useful relaxations!
- What happens here?

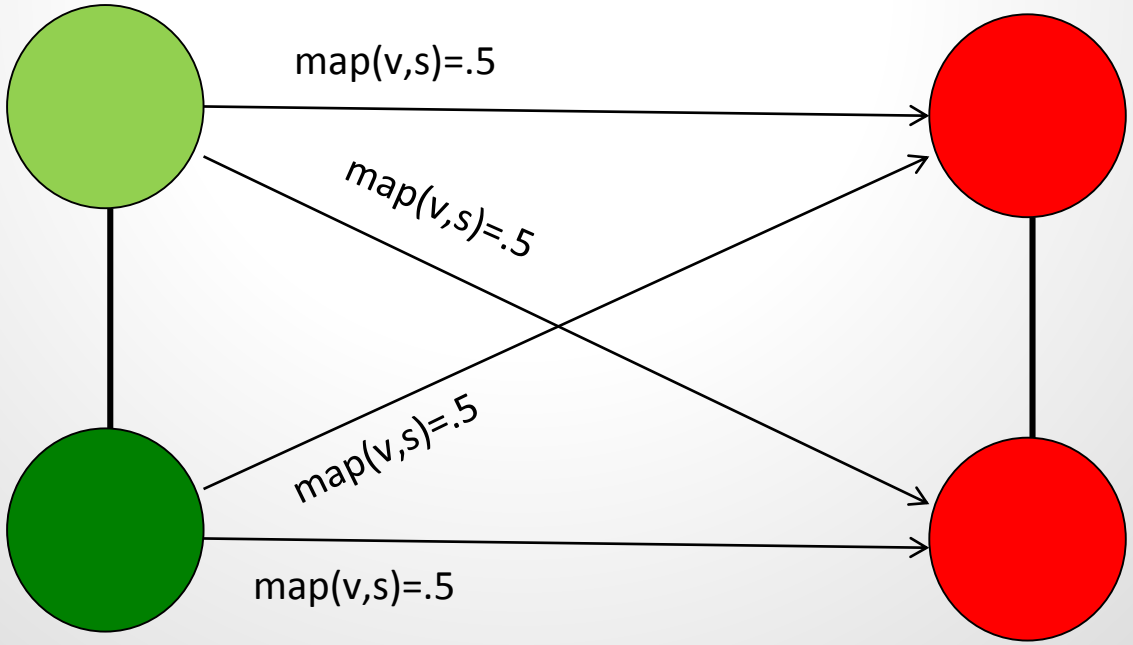
VNet:

Physical Network:

# Mixed Integer Programs (3)

- Recall: Relaxations useful if they give good bounds
- However it's hard to formulate a MIP for VNEP which yields useful relaxations!
- What happens here?

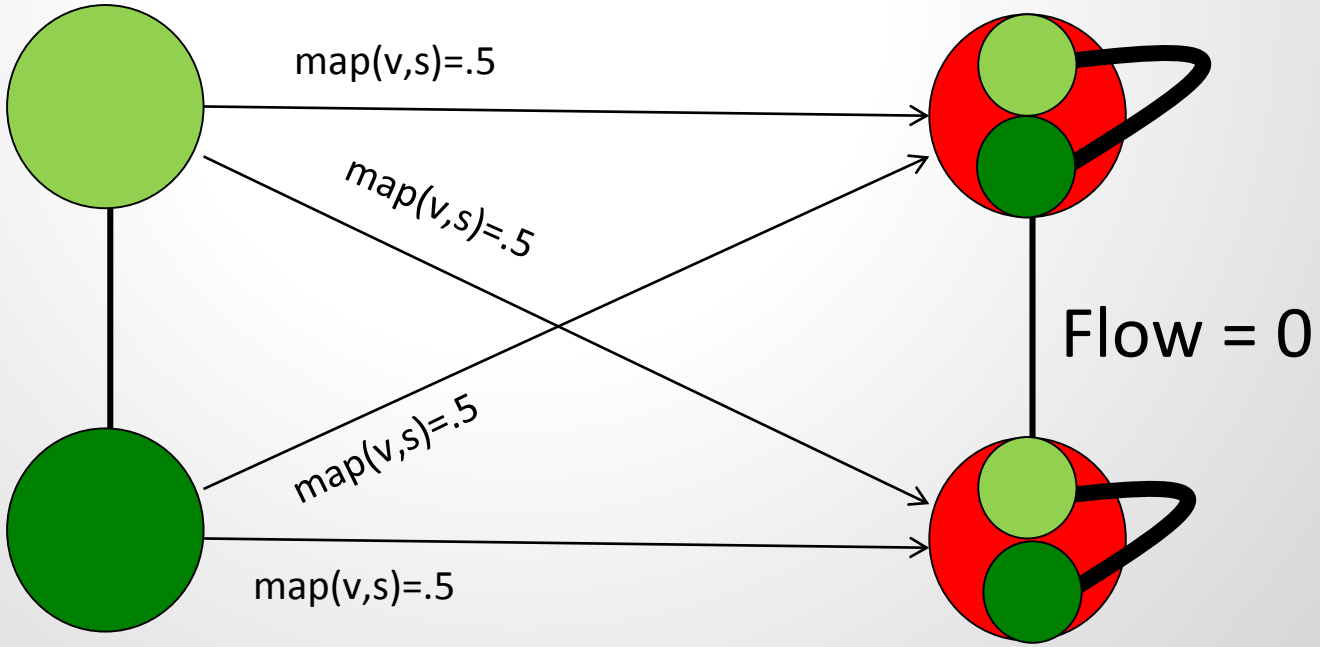
VNet: Physical Network:



# Mixed Integer Programs (3)

- Recall: Relaxations useful if they give good bounds
- However it's hard to formulate a MIP for VNEP which yields useful relaxations!
- What happens here?

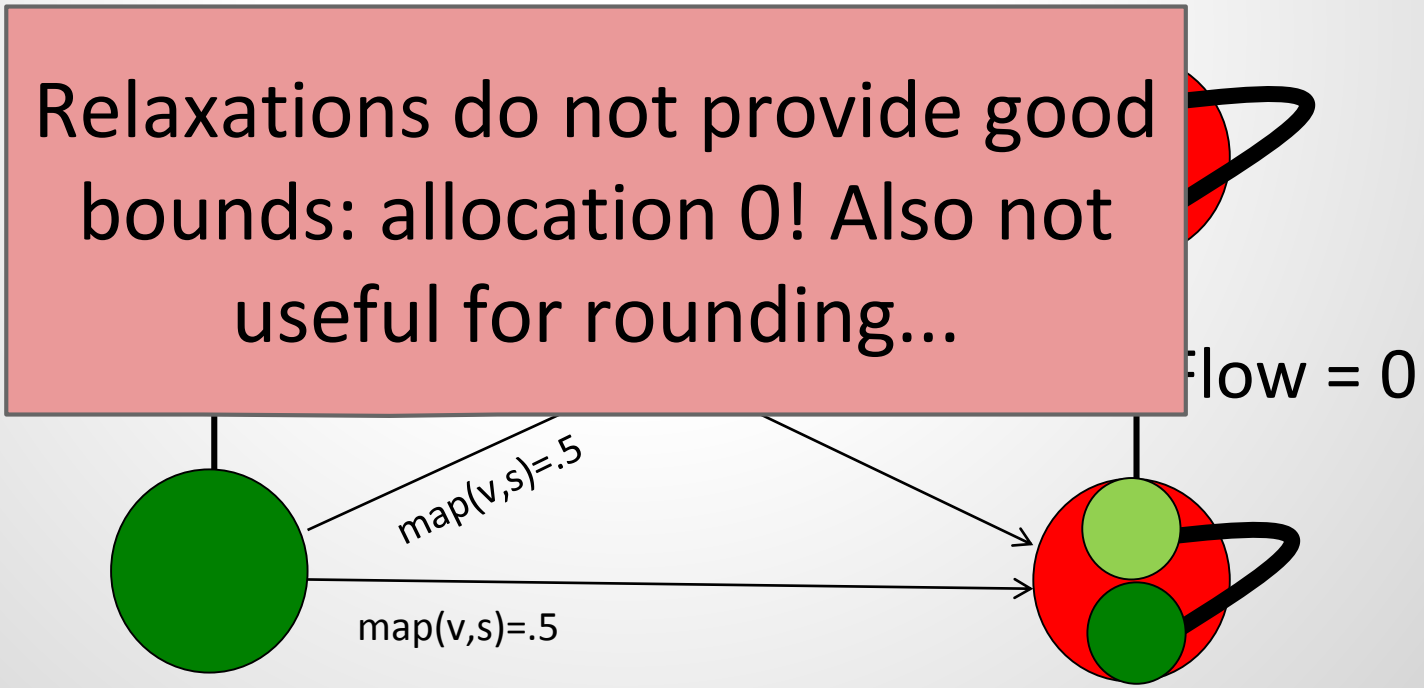
VNet: Physical Network:



# Mixed Integer Programs (3)

- Recall: Relaxations useful if they give good bounds
- However it's hard to formulate a MIP for VNEP which yields useful relaxations!
- What happens here?

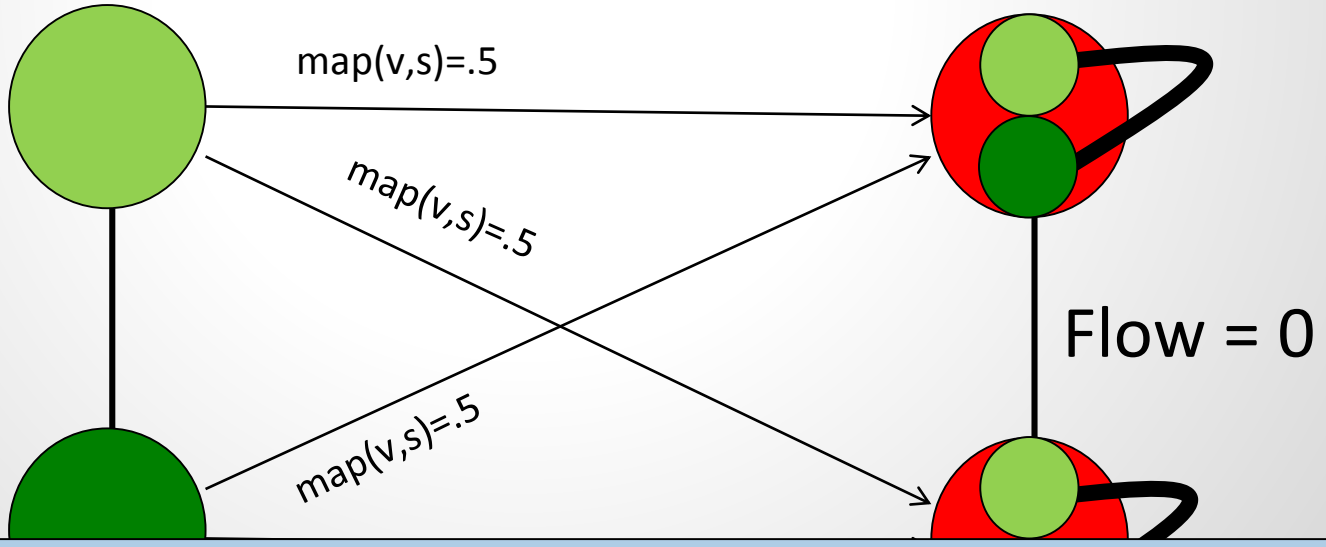
VNet: Physical Network:



# Mixed Integer Programs (3)

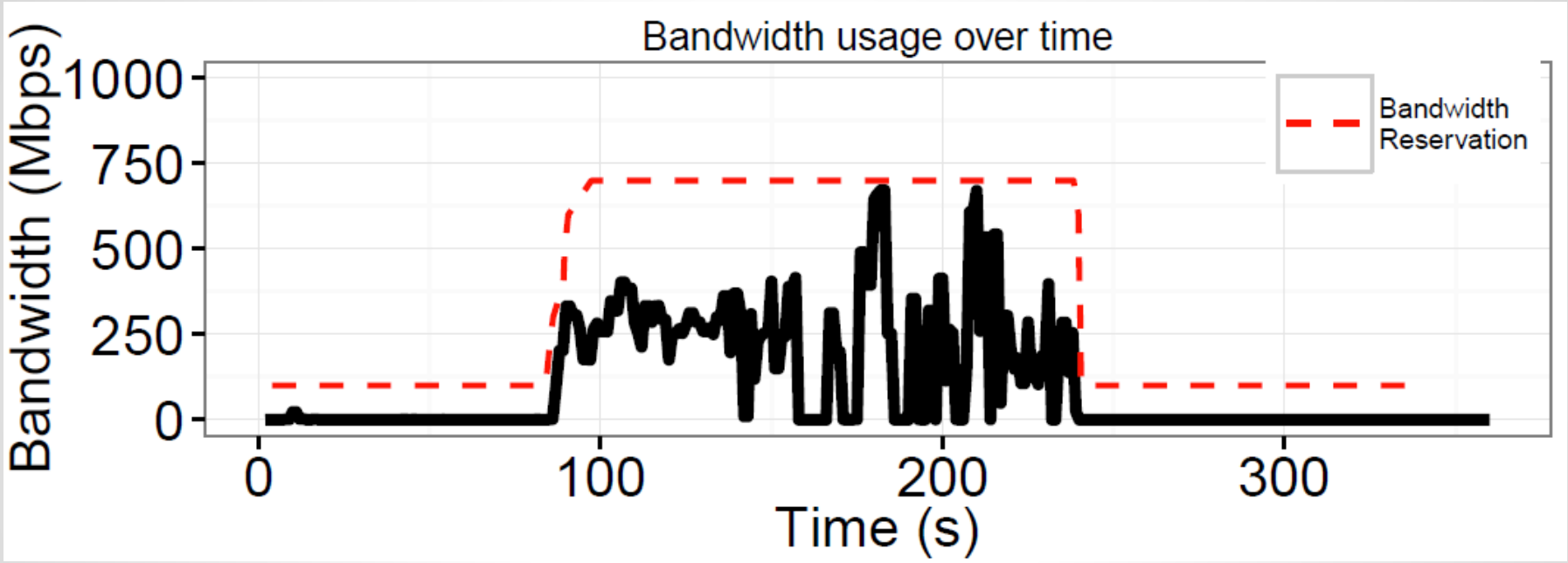
- Recall: Relaxations useful if they give good bounds
- However it's hard to formulate a MIP for VNEP which yields useful relaxations!
- What happens here?

VNet: Physical Network:




It's About Time: On Optimal Virtual Network Embeddings under Temporal Flexibilities
Matthias Rost, Stefan Schmid, and Anja Feldmann.
28th IEEE International Parallel and Distributed Processing Symposium (**IPDPS**), Phoenix, Arizona, USA, May 2014.

# Fixed Reservations Are Wasteful!



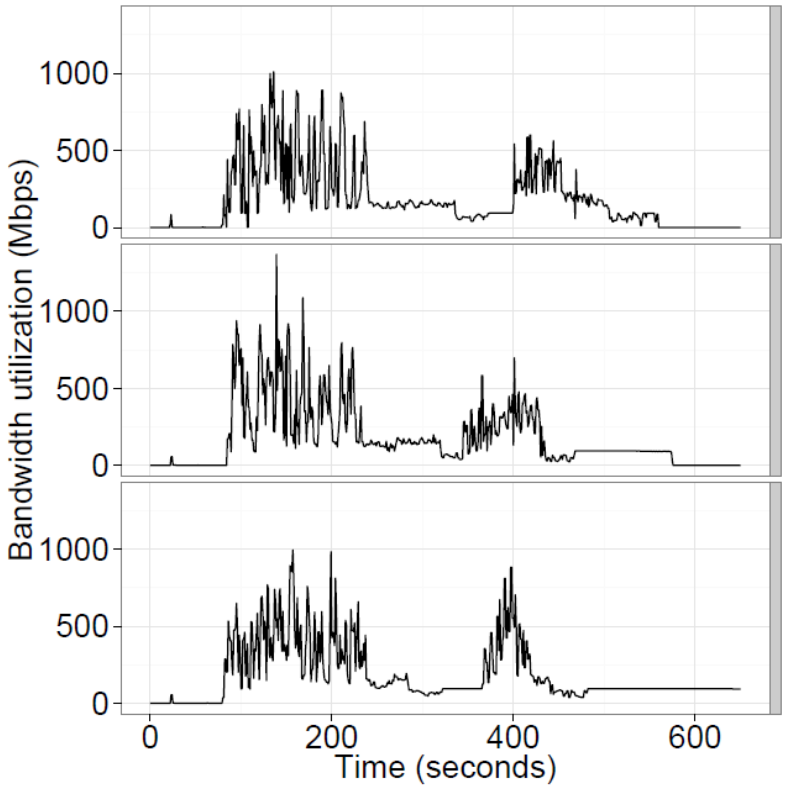

Bandwidth utilization of a TeraSort job over time.

In red: **Kraken**'s bandwidth reservation.

(Tasks inform Hadoop controller prior to shuffle phase; reservation with Linux *tc*.)

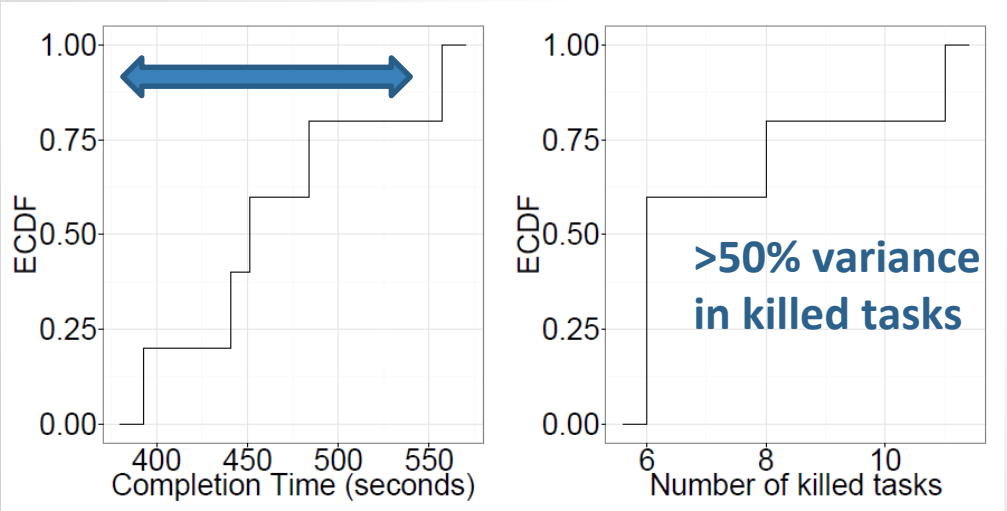
# Adaptive Reservations Are Hard to Predict! Need Online Adjustments *at Runtime*

- ***Temporal*** resource patterns are hard to predict
- Resource allocations must be changed ***online***




Bandwidth utilization of 3 different runs of the same **TeraSort workload (without interference)**

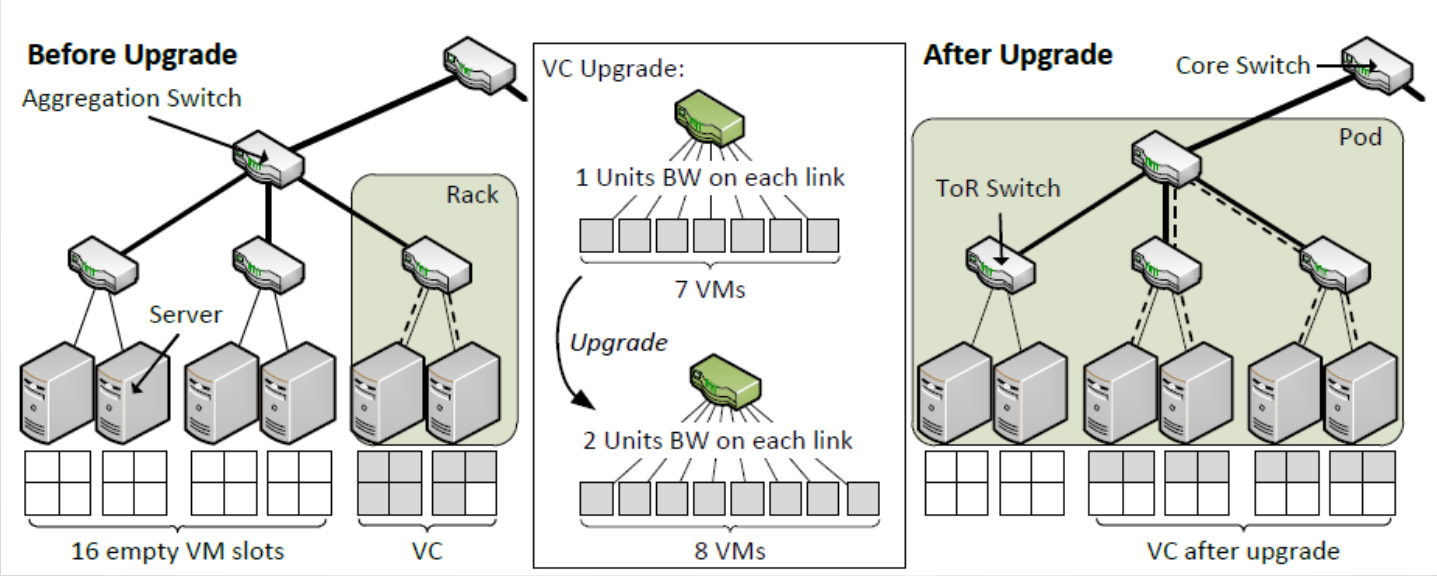
**>20% variance**




Completion times of jobs in the presence of **speculative execution** (*left*) and the number of speculated tasks (*right*)

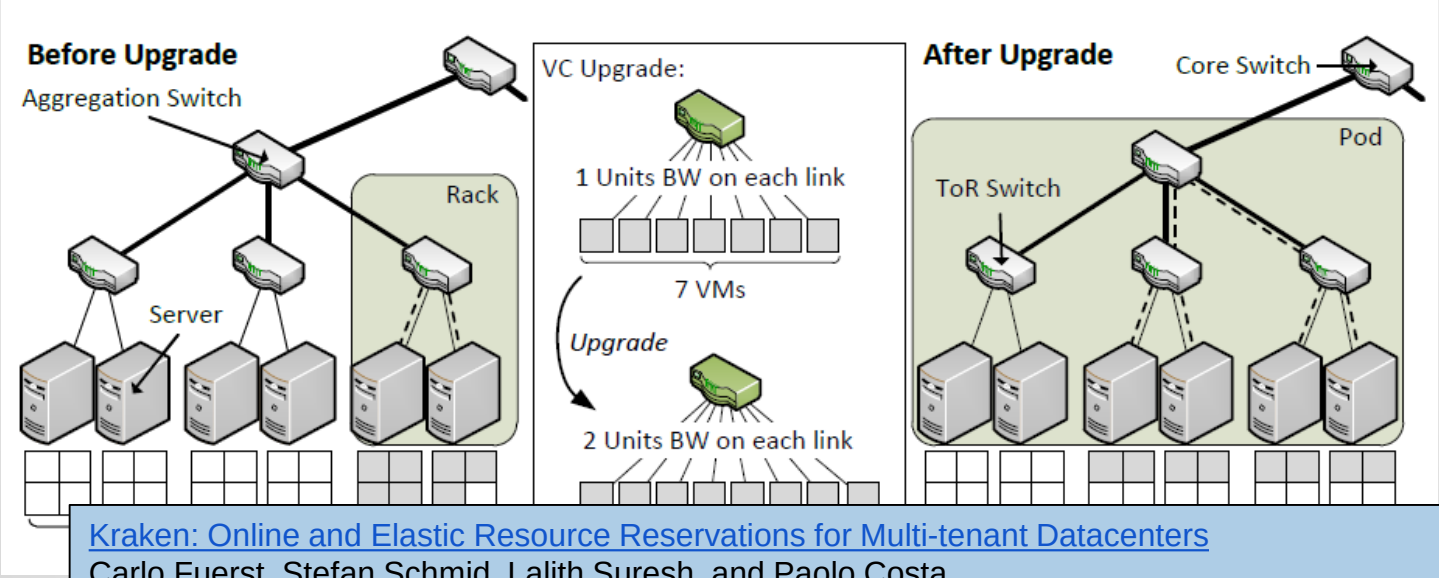
# Kraken: Elastic Reconfigurations

- Kraken:
  - Exploit flexibilities to deal with unpredictable execution
  - Scale up and down the virtual cluster at runtime
  - Supports task migrations
  - Currently for Hadoop only



# Kraken: Elastic Reconfigurations

- Kraken:
  - Exploit flexibilities to deal with unpredictable execution
  - Scale up and down the virtual cluster at runtime
  - Supports task migrations
  - Currently for Hadoop only




Kraken: Online and Elastic Resource Reservations for Multi-tenant Datacenters
Carlo Fuerst, Stefan Schmid, Lalith Suresh, and Paolo Costa.
35th IEEE Conference on Computer Communications (**INFOCOM**), San Francisco, California, USA, April 2016.

# Formal Guarantees Over Time

- How to provide guarantees over time?
- Realm of online algorithms and competitive analysis
  - Input to algorithm: sequence $\sigma$ (e.g., sequence of requests)
  - Online algorithm ON does not know requests t'>t
  - Needs to be perform close to optimal offline algorithm OFF who knows future!

**Competitive Analysis**

.

Competitive ratio $\rho$: max over all possible sequences $\sigma$

$\rho$ = Cost(ON)/Cost(OFF)

# Formal Guarantees Over Time

- How to provide guarantees over time?
- Realm of online algorithms and competitive analysis

**Nice:** If competitive ratio is low, there is no need to develop any sophisticated prediction models (which may be wrong anyway)! The guarantee holds in the worst-case.

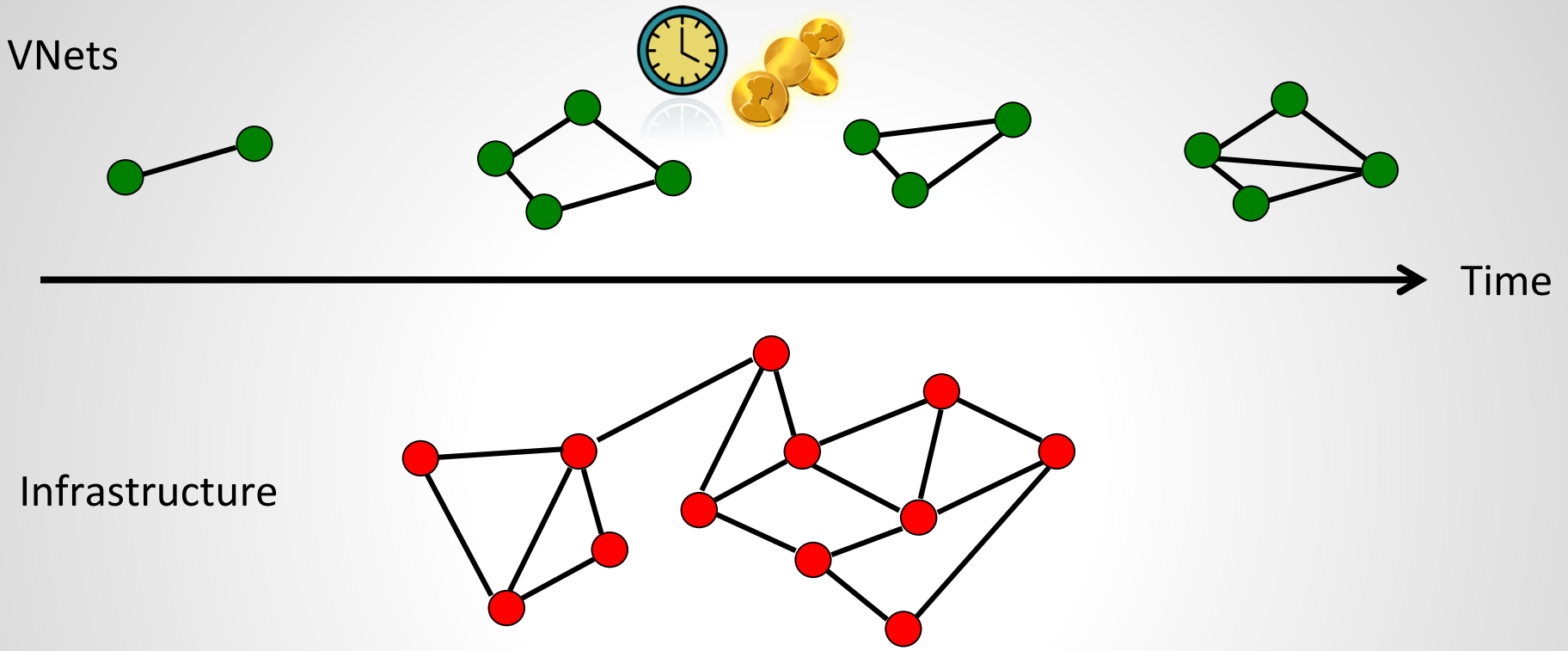
**Competitive Analysis**

.

Competitive ratio $\rho$: max over all possible sequences $\sigma$

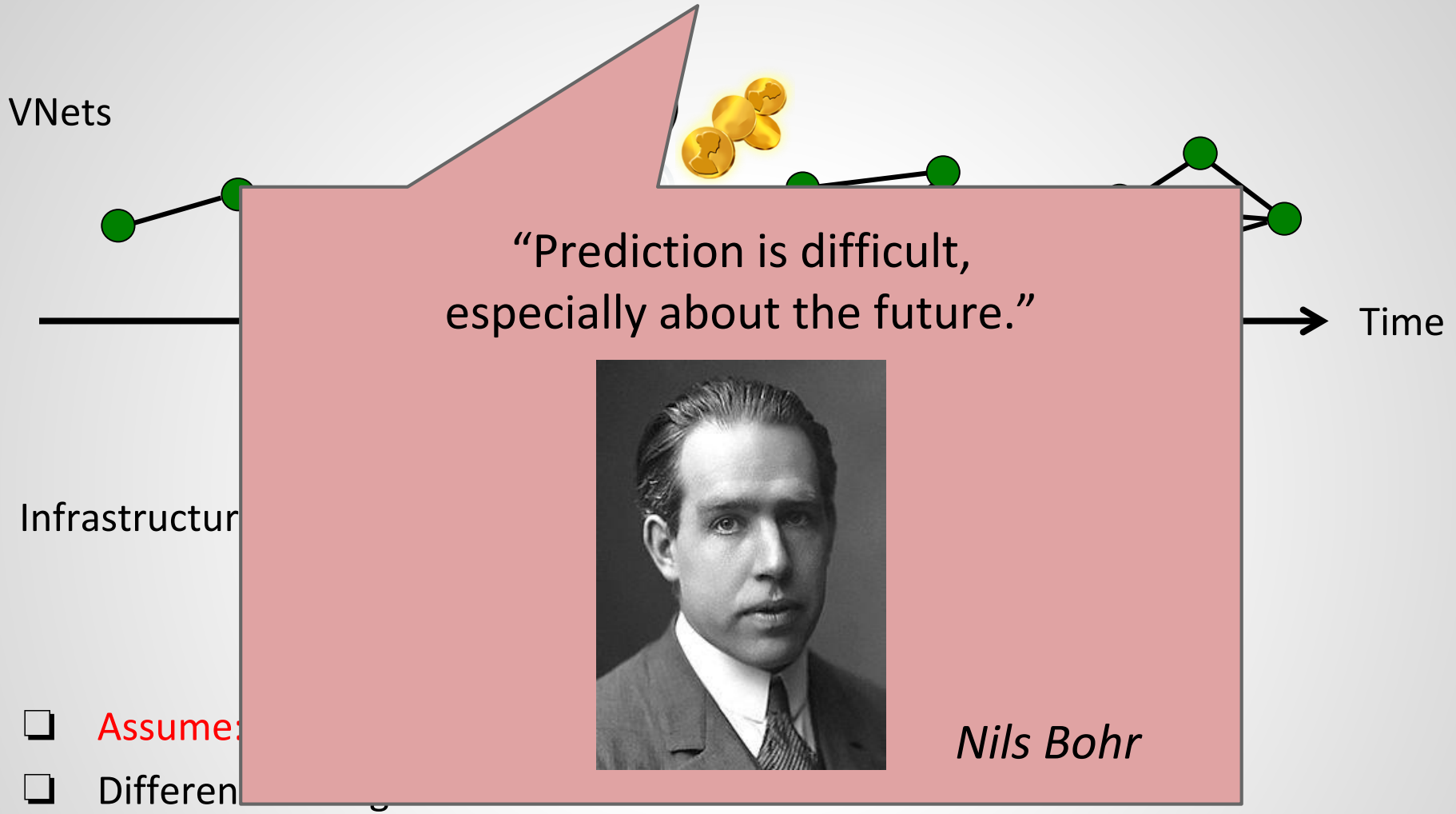
$\rho$ = Cost(ON)/Cost(OFF)

# Online Access Control (1)




- Assume: end-point locations given
- Different routing and traffic models
- Price and duration
- Which ones to accept?
- Online Primal-Dual Framework (Buchbinder and Naor)

# Online Access Control (1)



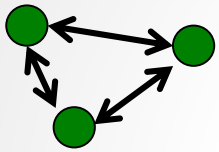

- ❑ Assume:
- ❑ Differen
- ❑ Price and duration
- ❑ Which ones to accept?
- ❑ Online Primal-Dual Framework (Buchbinder and Naor)

# Online Access Control (2)

- Traffic models

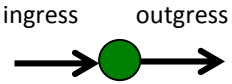
**Customer Pipe**

Traffic matrix: Bandwidth per VM pair (u,v)



**Hose Model**

Per VM bandwidth: polytope of traffic matrices.




virtual switch

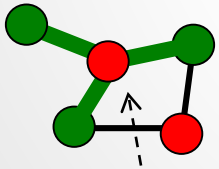
**Aggregate Ingress**

Only ingress specified: e.g., support multicast etc.


- Routing models

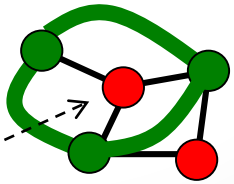
**Tree**

Steiner tree embedding



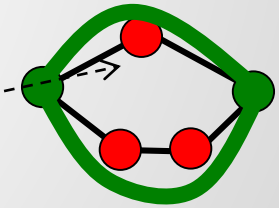
**Single Path**

Unsplittable paths



**Multi-Path**

Splittable paths (more capacity)



Relay costs: e.g., depending on packet rate

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$<br>$Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$<br>$X, Z_j \geq \mathbf{0}$ | $\max B_j^T \cdot Y_j \;\; s.t.$<br>$A_j \cdot Y_j \leq C$<br>$D_j \cdot Y_j \leq \mathbf{1}$<br>$Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$ | $\max B_j^T \cdot Y_j \;\; s.t.$ |
| $Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$ | $A_j \cdot Y_j \leq C$ |
| $X, Z_j \geq \mathbf{0}$ | $D_j \cdot Y_j \leq \mathbf{1}$ |
| | $Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

Formulate the packing (dual) LP: Maximize profit

(Note: dynamic LP!)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
| --- | --- |
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$<br>$Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$<br>$X, Z_j \geq \mathbf{0}$ | $\max B_j^T \cdot Y_j \;\; s.t.$<br>$A_j \cdot Y_j \leq C$<br>$D_j \cdot Y_j \leq \mathbf{1}$<br>$Y_j \geq \mathbf{0}$ |

s.t. constraints

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

Primal and Dual

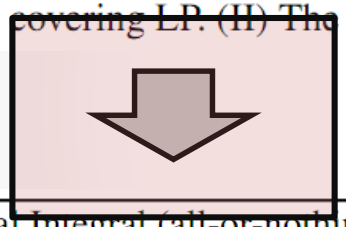
# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$ | $\max B_j^T \cdot Y_j \;\; s.t.$ |
| $Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$ | $A_j \cdot Y_j \leq C$ |
| $X, Z_j \geq \mathbf{0}$ | $D_j \cdot Y_j \leq \mathbf{1}$ |
| | $Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)



primal-dual framework

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$ | $\max B_j^T \cdot Y_j \;\; s.t.$ |
| $Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$ | $A_j \cdot Y_j \leq C$ |
| $X, Z_j \geq \mathbf{0}$ | $D_j \cdot Y_j \leq \mathbf{1}$ |
| | $Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure) ← optimal embedding!
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$ | $\max B_j^T \cdot Y_j \;\; s.t.$ |
| $Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$ | $A_j \cdot Y_j \leq C$ |
| $X, Z_j \geq \mathbf{0}$ | $D_j \cdot Y_j \leq \mathbf{1}$ |
| | $Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

Embedding cost vs profit?

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$ | $\max B_j^T \cdot Y_j \;\; s.t.$ |
| $Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$ | $A_j \cdot Y_j \leq C$ |
| $X, Z_j \geq \mathbf{0}$ | $D_j \cdot Y_j \leq \mathbf{1}$ |
| | $Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

If cheap: accept and update primal variables (always feasible solution)

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$<br>$Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$<br>$X, Z_j \geq \mathbf{0}$ | $\max B_j^T \cdot Y_j \;\; s.t.$<br>$A_j \cdot Y_j \leq C$<br>$D_j \cdot Y_j \leq \mathbf{1}$<br>$Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

Else reject

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$ | $\max B_j^T \cdot Y_j \;\; s.t.$ |
| $Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$ | $A_j \cdot Y_j \leq C$ |
| $X, Z_j \geq \mathbf{0}$ | $D_j \cdot Y_j \leq \mathbf{1}$ |
| | $Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

Computationally hard!

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$ | $\max B_j^T \cdot Y_j \;\; s.t.$ |
| $Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$ | $A_j \cdot Y_j \leq C$ |
| $X, Z_j \geq \mathbf{0}$ | $D_j \cdot Y_j \leq \mathbf{1}$ |
| | $Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

Computationally hard!

Use your favorite approximation algorithm! If competitive ratio ρ and approximation r, overall competitive ratio ρ*r.

**Primal and Dual**

# Online Access Control (3)

| (I) | (II) |
|---|---|
| $\min Z_j^T \cdot \mathbf{1} + X^T \cdot C \;\; s.t.$ | $\max B_j^T \cdot Y_j \;\; s.t.$ |
| $Z_j^T \cdot D_j + X^T \cdot A_j \geq B_j^T$ | $A_j \cdot Y_j \leq C$ |
| $X, Z_j \geq \mathbf{0}$ | $D_j \cdot Y_j \leq \mathbf{1}$ |
| | $Y_j \geq \mathbf{0}$ |

Fig. 1: (I) The primal covering LP. (II) The dual packing LP.

**Competitive Analysis**

Does not know t'>t.
Competitive ratio:
r = Cost(ON)/Cost(OFF)

**Algorithm**

**Algorithm 1** The General Integral (all-or-nothing) Packing Online Algorithm (GIPO).

Upon the $j$th round:

1. $f_{j,\ell} \leftarrow \text{argmin}\{\gamma(j,\ell) : f_{j,\ell} \in \Delta_j\}$ (oracle procedure)
2. If $\gamma(j,\ell) < b_j$ then, (accept)
   (a) $y_{j,\ell} \leftarrow 1$.
   (b) For each row $e$ : If $A_{e,(j,\ell)} \neq 0$ do

$$x_e \leftarrow x_e \cdot 2^{A_{e,(j,\ell)}/c_e} + \frac{1}{w(j,\ell)} \cdot (2^{A_{e,(j,\ell)}/c_e} - 1).$$

   (c) $z_j \leftarrow b_j - \gamma(j,\ell)$.
3. Else, (reject)
   (a) $z_j \leftarrow 0$.

Competitive and Deterministic Embeddings of Virtual Networks
Guy Even, Moti Medina, Gregor Schaffrath, and Stefan Schmid.
Journal Theoretical Computer Science (**TCS**), Elsevier, 2013..

# Challenges of More Flexible Networked Systems

1. Kraken: Predictable cloud application performance through adaptive virtual clusters

2. **C3: Low tail latency in cloud data stores through replica selection**

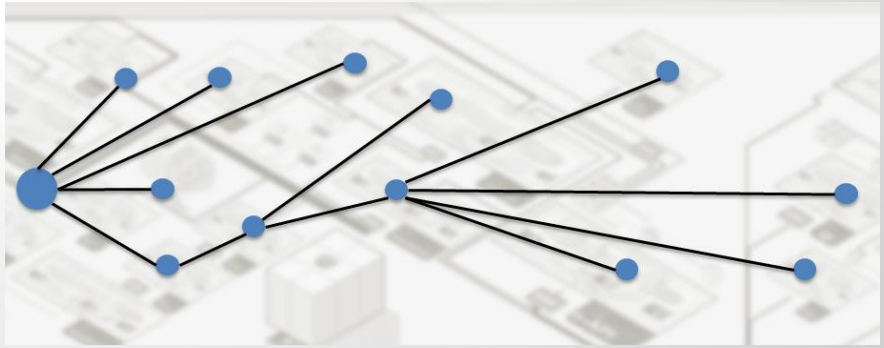
3. Peacock: Consistent network updates

4. Panopticon: How to introduce these innovative technologies in the first place? Case study: SDN

# What about predictable latency?

- ***Tail latency:*** Performance challenge even in well-provisioned systems
  - **Skews** in demand, time-varying service times, stragglers, ...
  - No time to make make **rigorous optimizations or reservations**

- **Tail** matters...
  - Today's interactive **web** applications require **fluid** response time
  - Degraded user experience directly impacts **revenue**
  - Web applications = multi-tier, **large** distributed systems
  - 1 request involves **10(0)s** data accesses / servers!



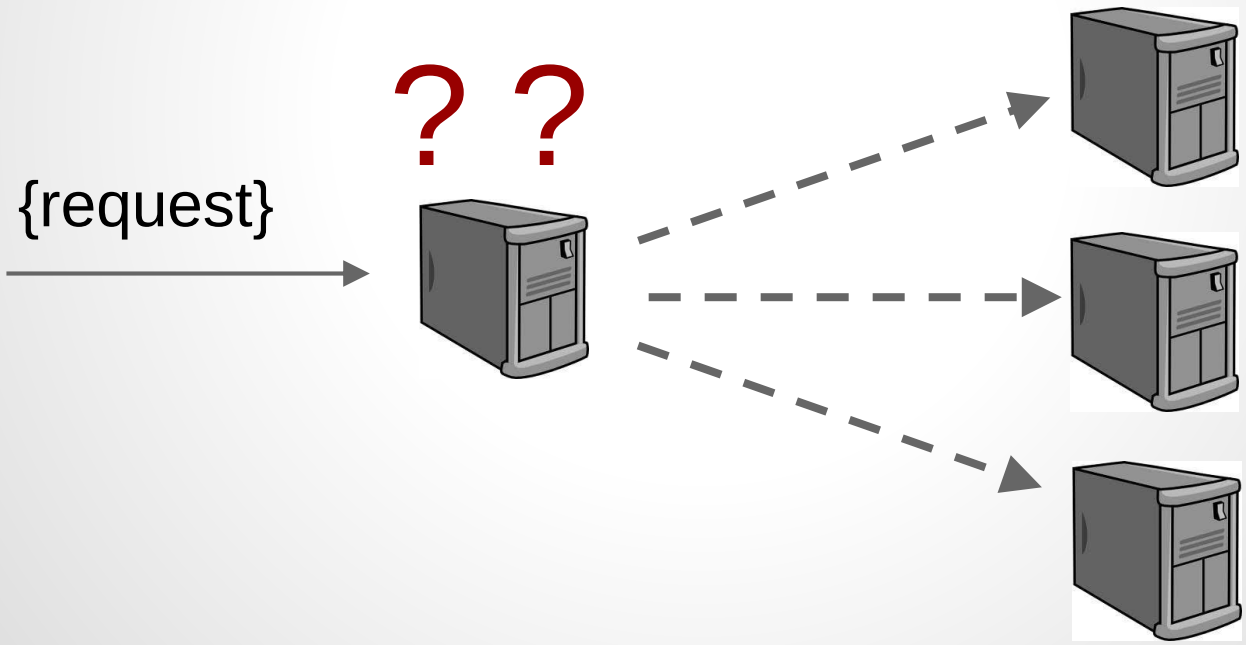
# How to cut tail latency?

- Idea C3: Exploit **replica selection**!
  - Many distributed DBs resp. **key-value stores** feature redundancy
  - **Opportunity** often overlooked so far

- Our focus: **Cassandra** (1-hop DHT, server = client)
  - Powers, e.g., Ebay, Netflix, Spotify
  - More sophisticated than MongoDB or Riak

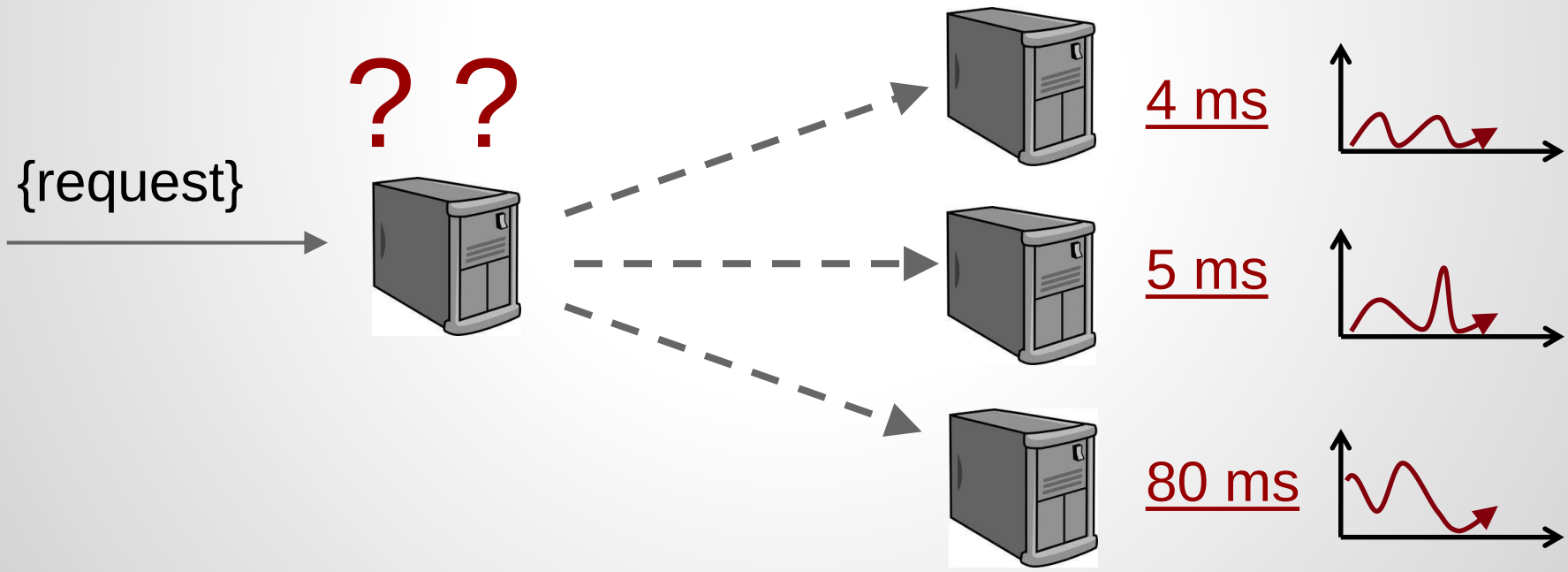
# C3: Exploit Replica Selection

- Great idea! But how? Just go for «the best»?



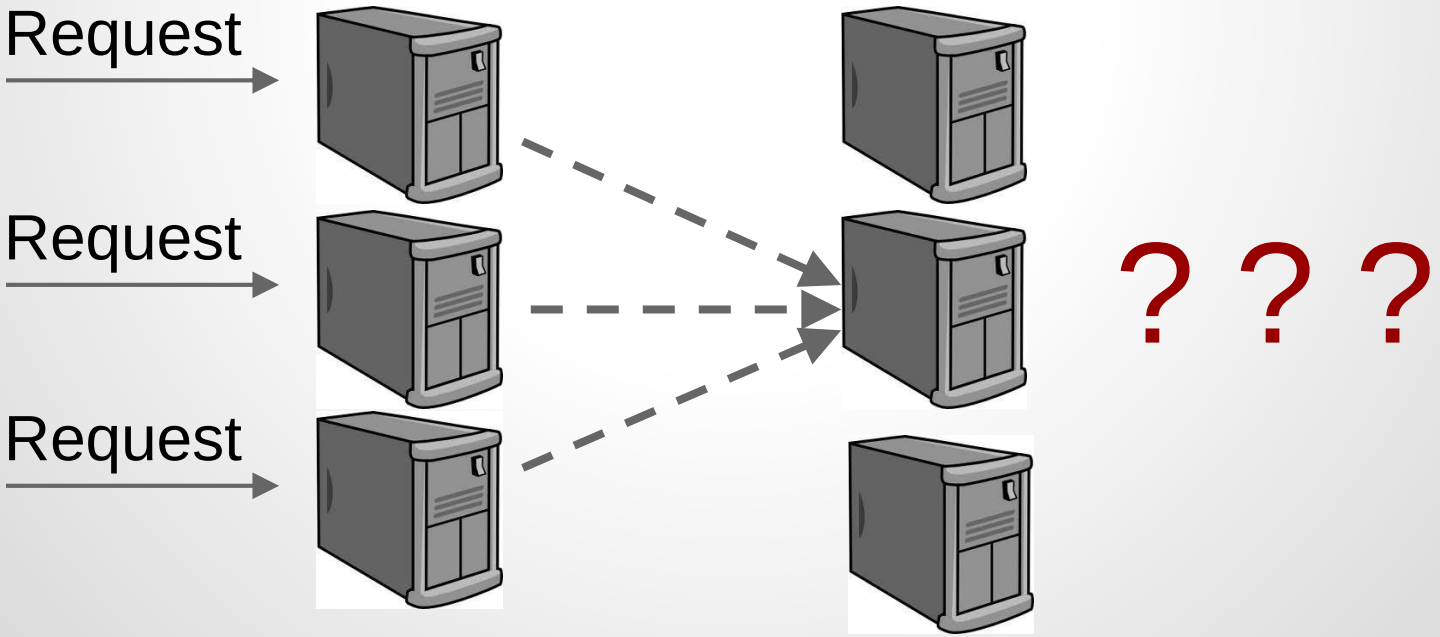
# Careful: «The best» can change

- Not so simple!
  - Need to deal with **heterogenous** and **time-varying** service times
  - Background garbage collection, log compaction, TCP, deamons



# Careful: Herd Behavior

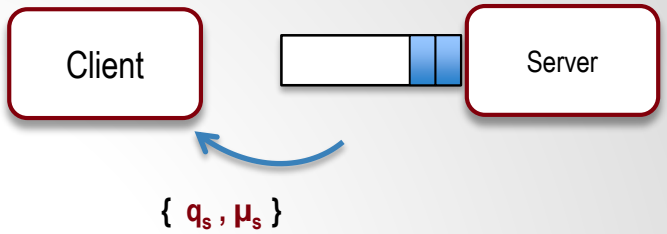
- Potentially high **fan-in** and **herd behavior**!
- Observed in Cassandra Dynamic Snitching (DS)
  - Coarse **time intervals** and **I/O gossiping**
  - **Synchronization** and stale information



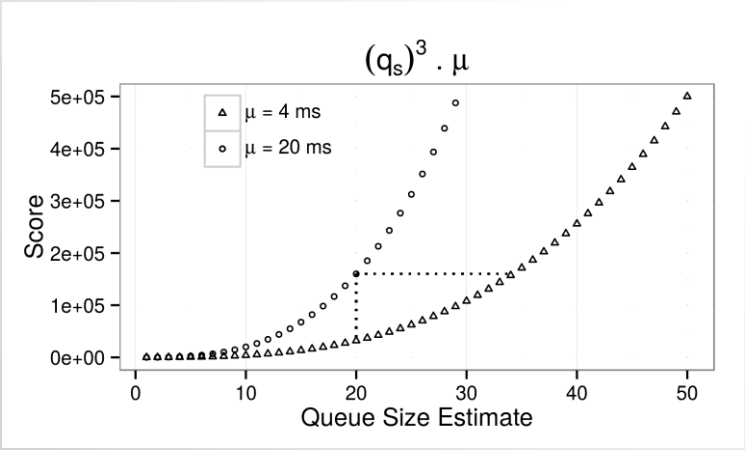

**A coordination / control theory problem!**

# C3 in a Nutshell

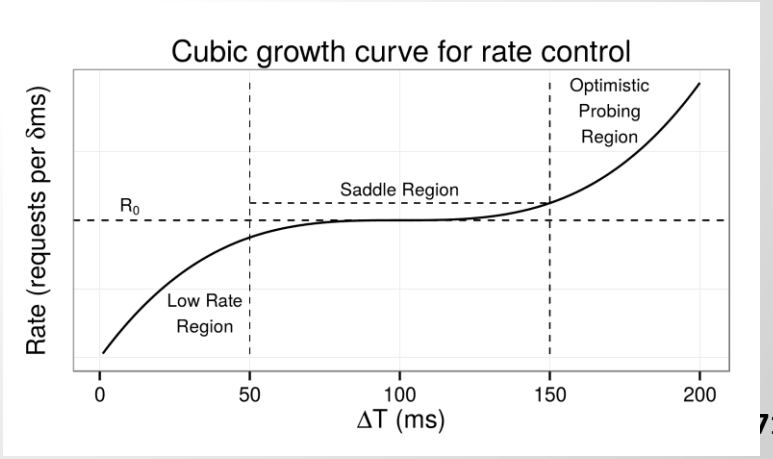
- 4 Principles:
  - Stay informed: **piggy-back** queue state and service times
  - Stay reactive and don't commit: use **backpressure queue**
  - Leverage heterogeity: **compensate** for service times
  - Avoid redundancy




- Mechanism 1: replica ranking
  - Penalize larger queues



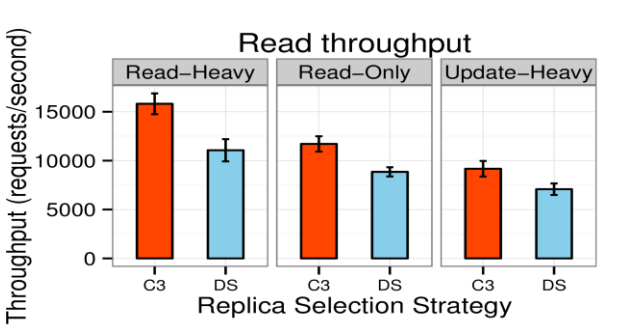

- Mechanism 2: rate control
  - Goal: match service rate and keep pipeline full
  - Cubic, with saddle region



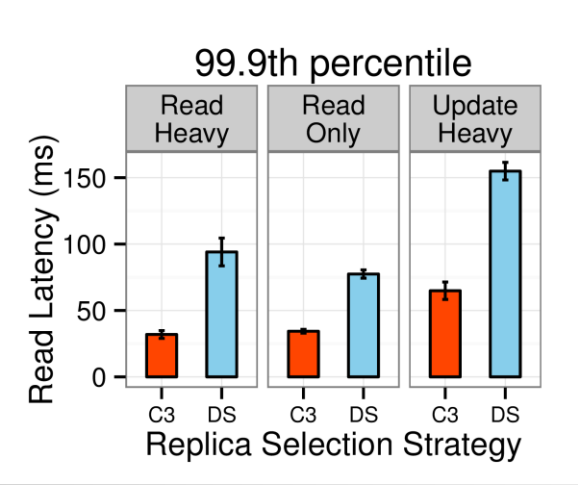
# Performance Evaluation

- Methodology:
  - **Amazon EC2**
    - disk vs SSD
  - BigFoot testbed
  - Simulations

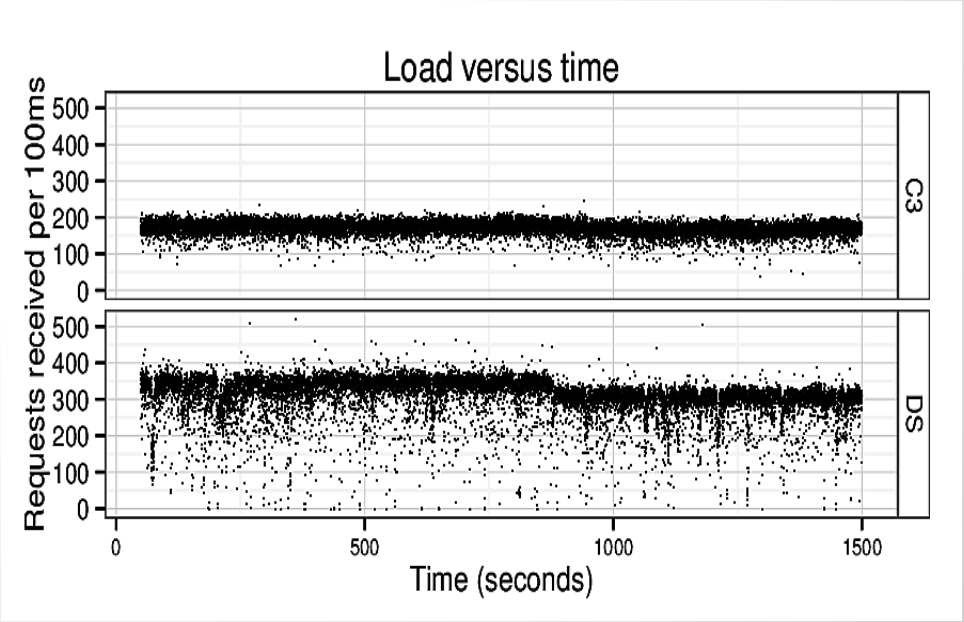
- Higher read throughput...




- Lower tail latency
  - 2-3x for 99.9%



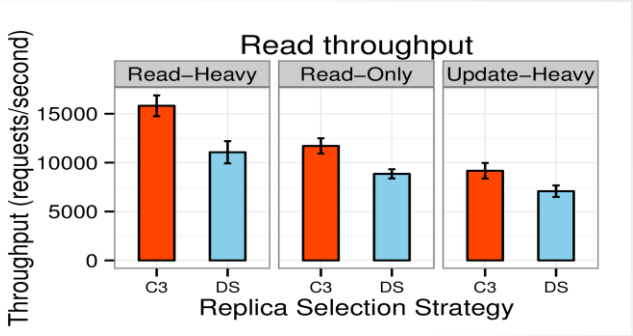

- ... and lower load (and variance)!



# Performance Evaluation

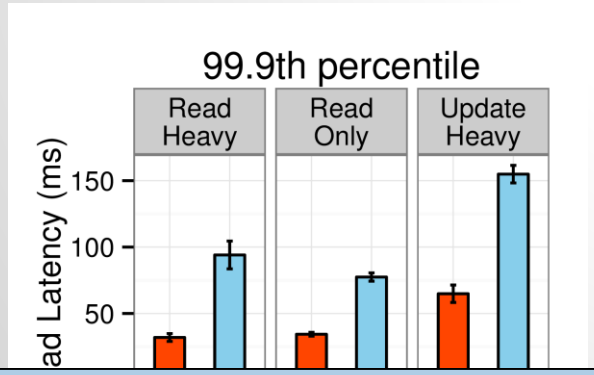
- Methodology:
  - **Amazon EC2**
    - disk vs SSD
  - BigFoot testbed
  - Simulations

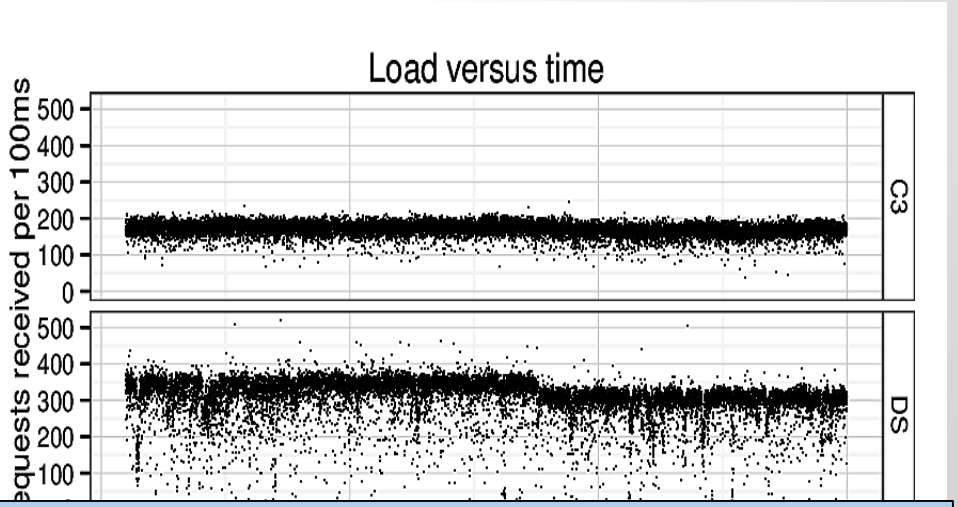
- Higher read throughput...




- Lower tail latency
  - 2-3x for 99.9%

- ... and lower load (and variance)!




C3: Cutting Tail Latency in Cloud Data Stores via Adaptive Replica Selection
Lalith Suresh, Marco Canini, Stefan Schmid, and Anja Feldmann.
12th USENIX Symposium on Networked Systems Design and Implementation (**NSDI**), Oakland, California, USA, May 2015.

# Challenges of More Flexible Networked Systems

1. Kraken: Predictable cloud application performance through adaptive virtual clusters

2. C3: Low tail latency in cloud data stores through replica selection

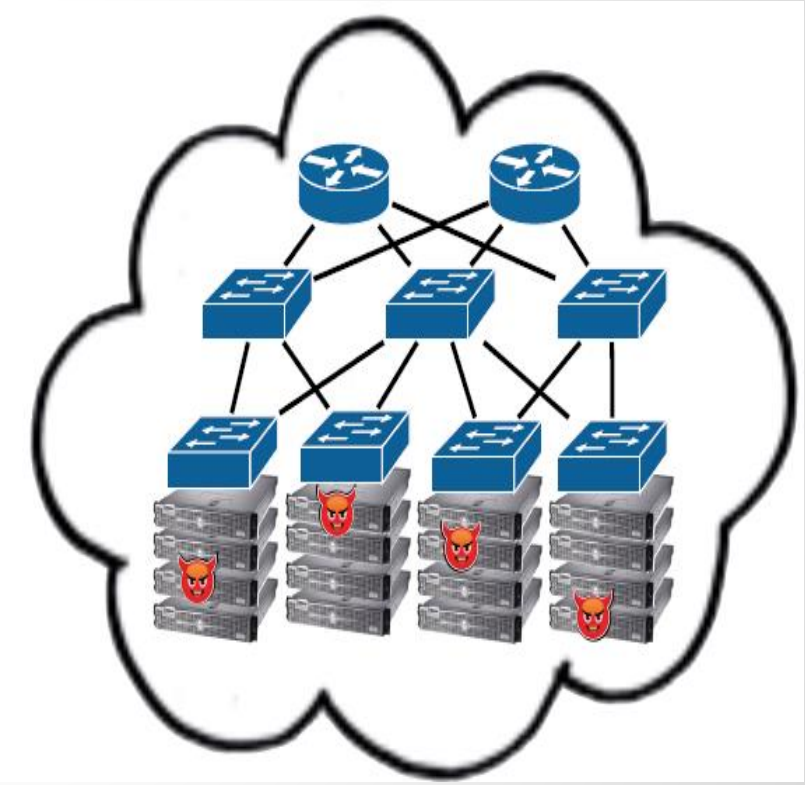
3. **Peacock: Consistent network updates**

4. Panopticon: How to introduce these innovative technologies in the first place? Case study: SDN

# Why Consistency Matters

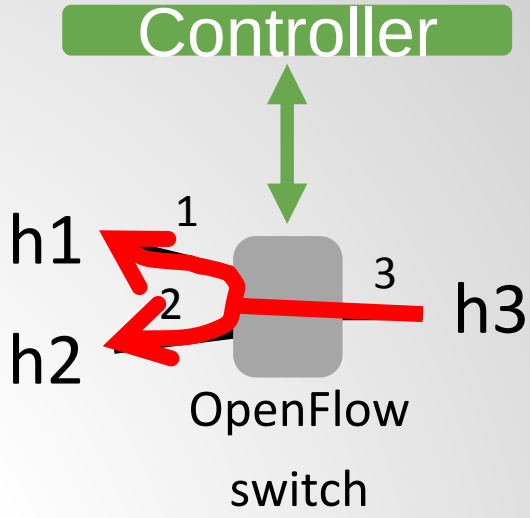
Important, e.g., in Cloud

What if your traffic was *not* isolated from other tenants during periods of routine maintenance?



Thanks to Nate Foster for example!

# Jennifer Rexford's Example: SDN MAC Learning Done Wrong




- MAC learning: The «Hello World»
  - a bug in early controller versions
- In legacy networks simple
  - Flood packets sent to unknown destinations
  - Learn host's location when it sends packets
- Pitfalls in SDN: learn sender => miss response
  - Assume: low priority rule * (no match): send to controller
  - h1->h2: Add rule h1@port1 (location learned)
  - Controller misses h2->h1 (as h1 known, h2 stay unknown!)
  - When h3->h2: flooding forever (learns h3, never learns h2)

Thanks to Jen Rexford for example!

# Example: Outages

Even technically sophisticated companies are struggling to build networks that provide reliable performance.


*We discovered a misconfiguration on this pair of switches that caused what's called a "bridge loop" in the network.*

*A network change was […] executed incorrectly […] more "stuck" volumes and added more requests to the re-mirroring storm*

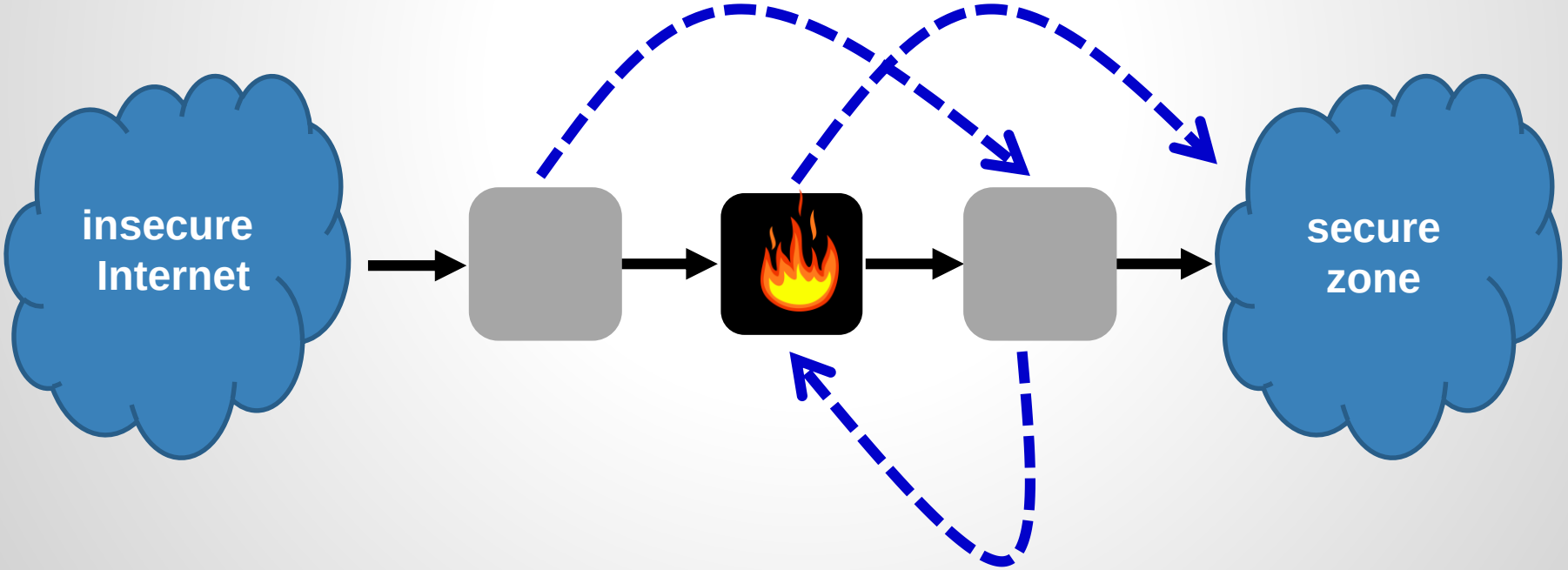


*Service outage was due to a series of internal network events that corrupted router data tables*

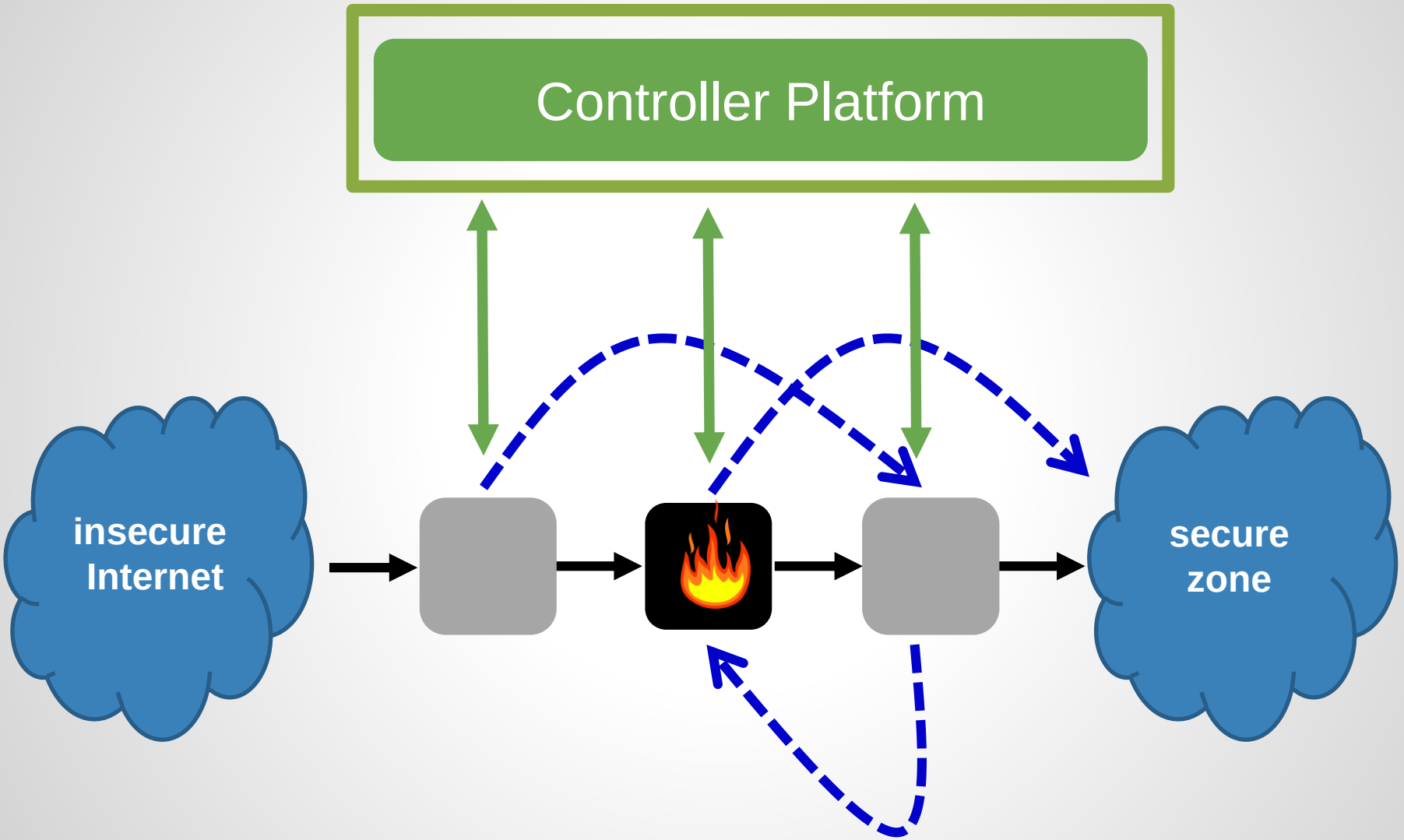
*Experienced a network connectivity issue […] interrupted the airline's flight departures, airport processing and reservations systems*


Thanks to Nate Foster for examples (at DSDN 2014)!

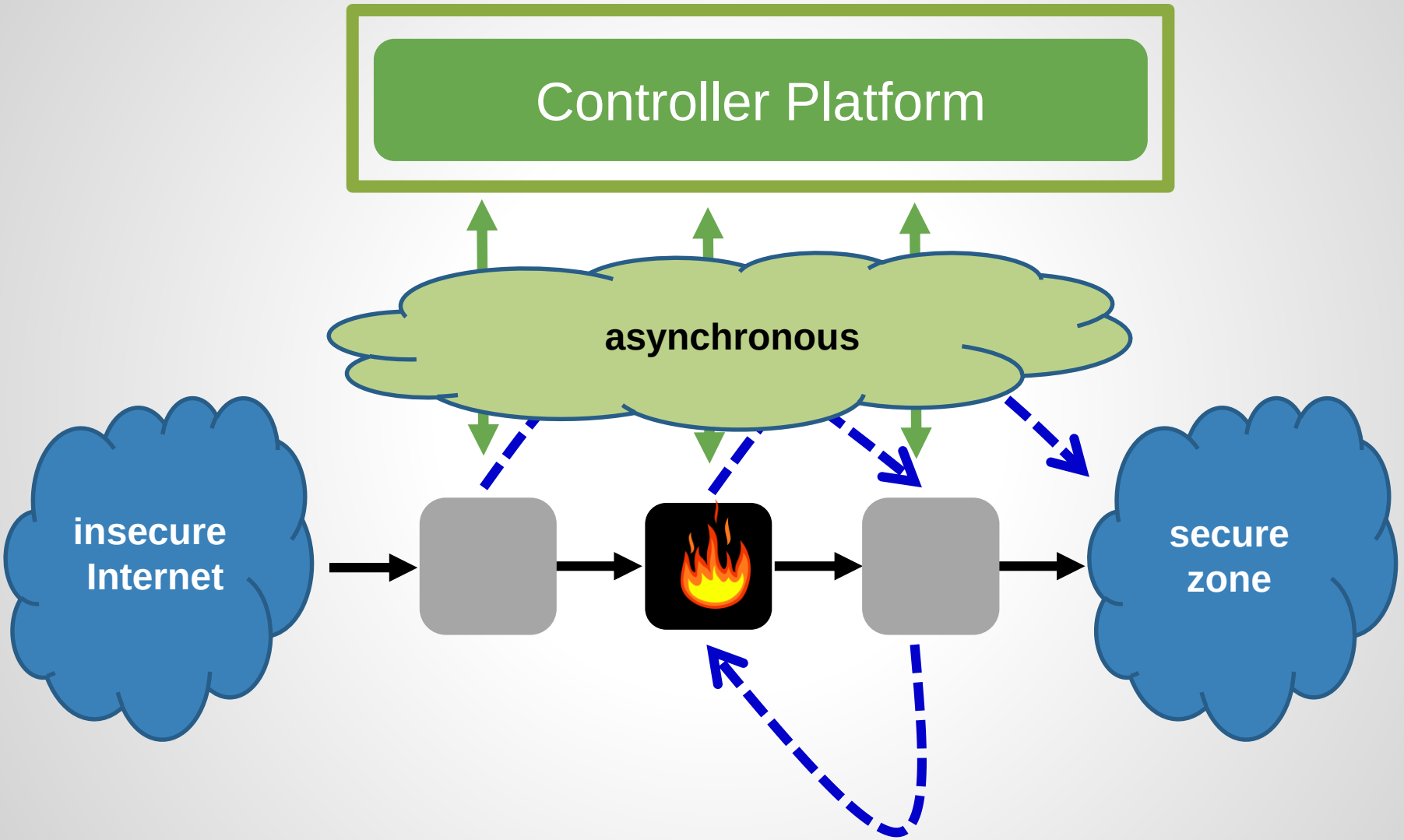
# Challenge: Multi-Switch Updates



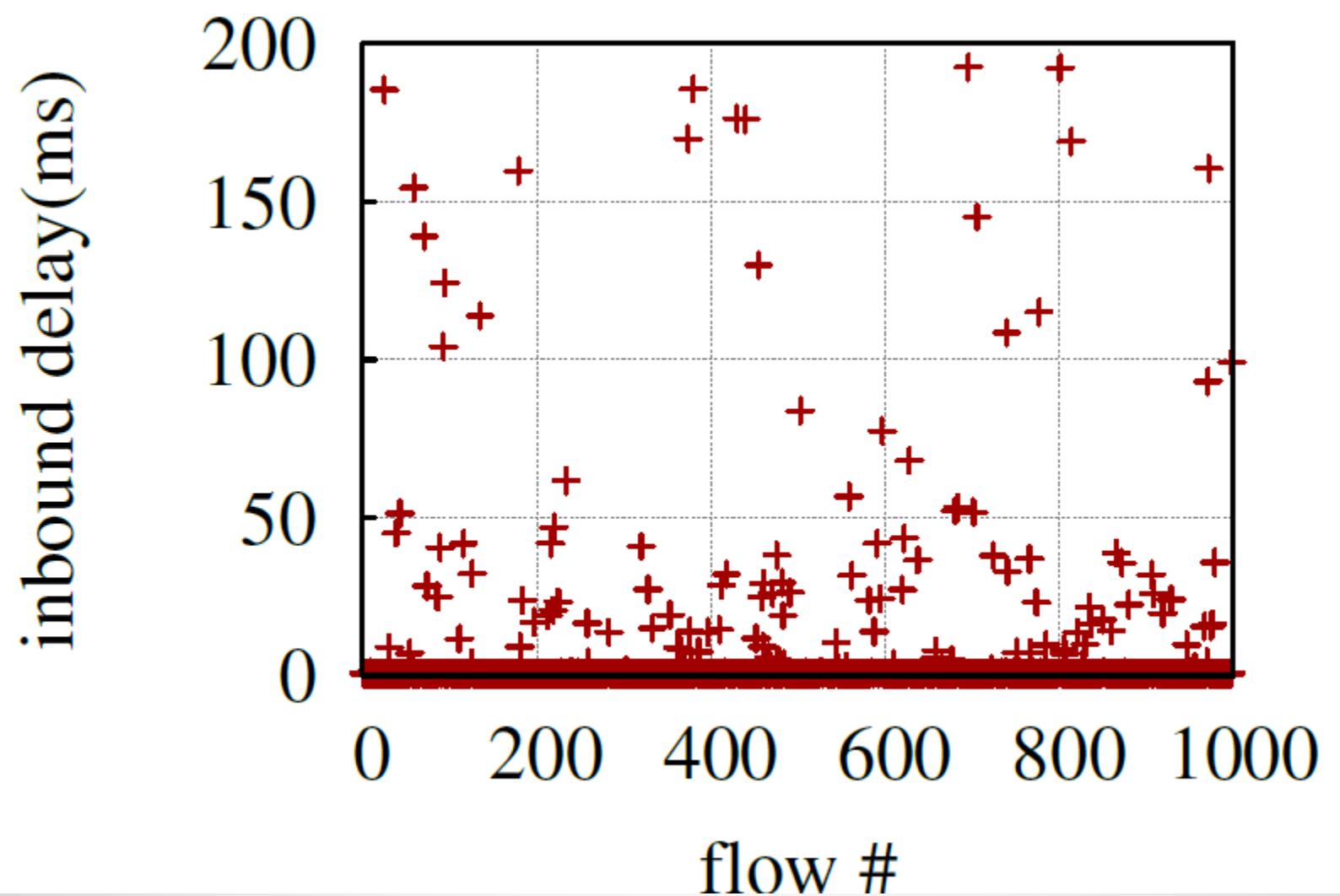
# Challenge: Multi-Switch Updates



# Challenge: Multi-Switch Updates



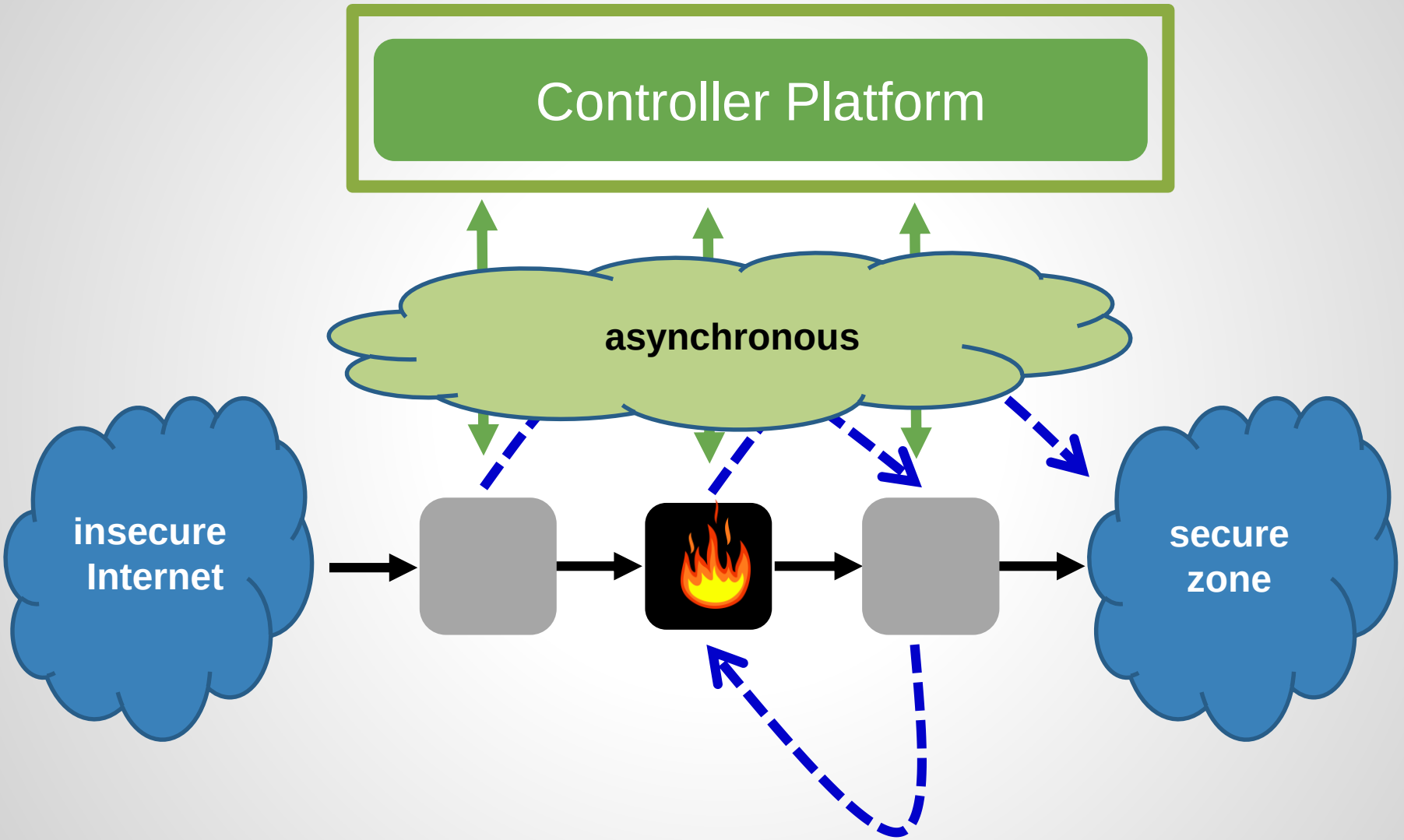
# An Asynchronous Distributed System



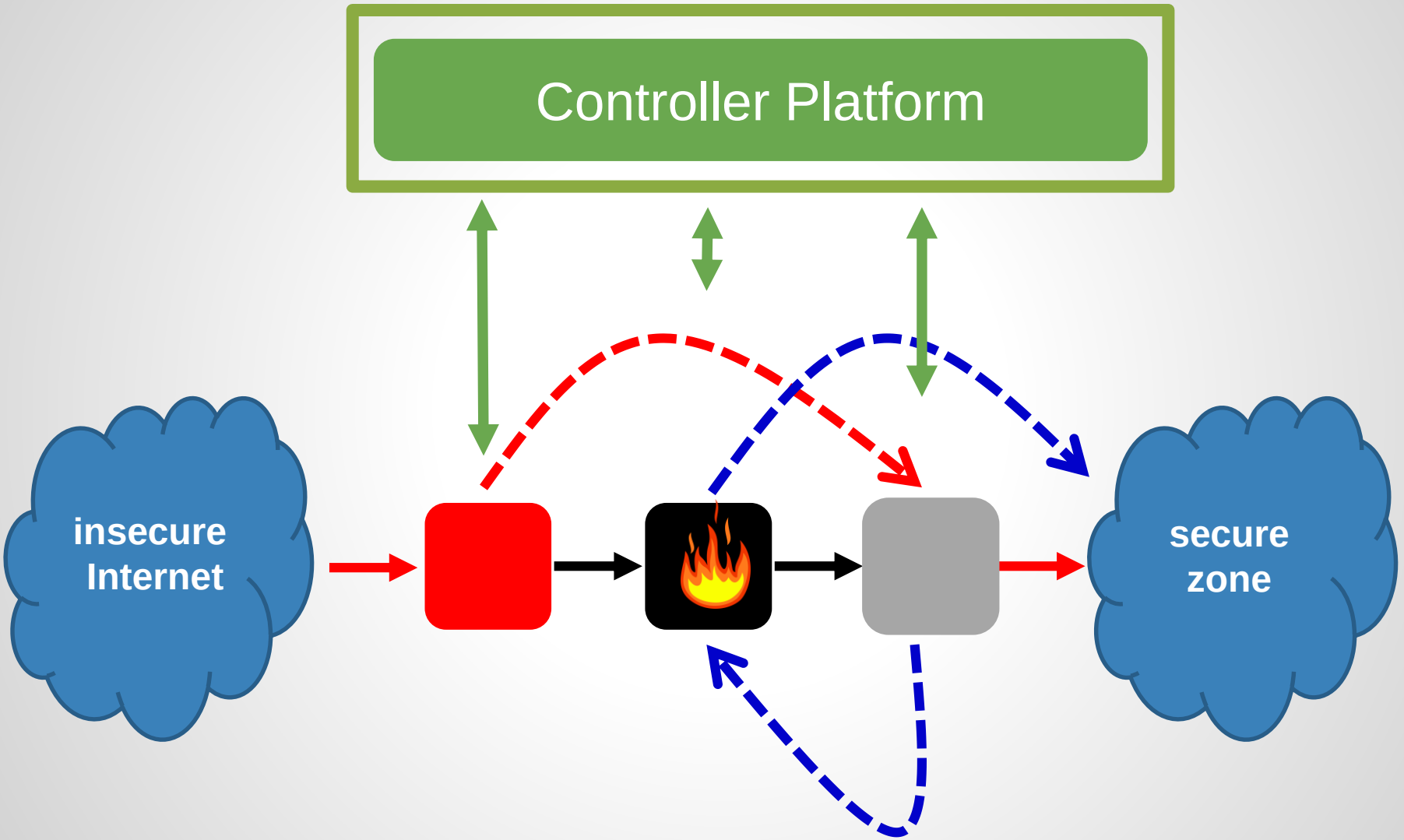

He et al., ACM SOSR 2015: without network latency

Jin et al., ACM SIGCOMM 2014: even higher variance

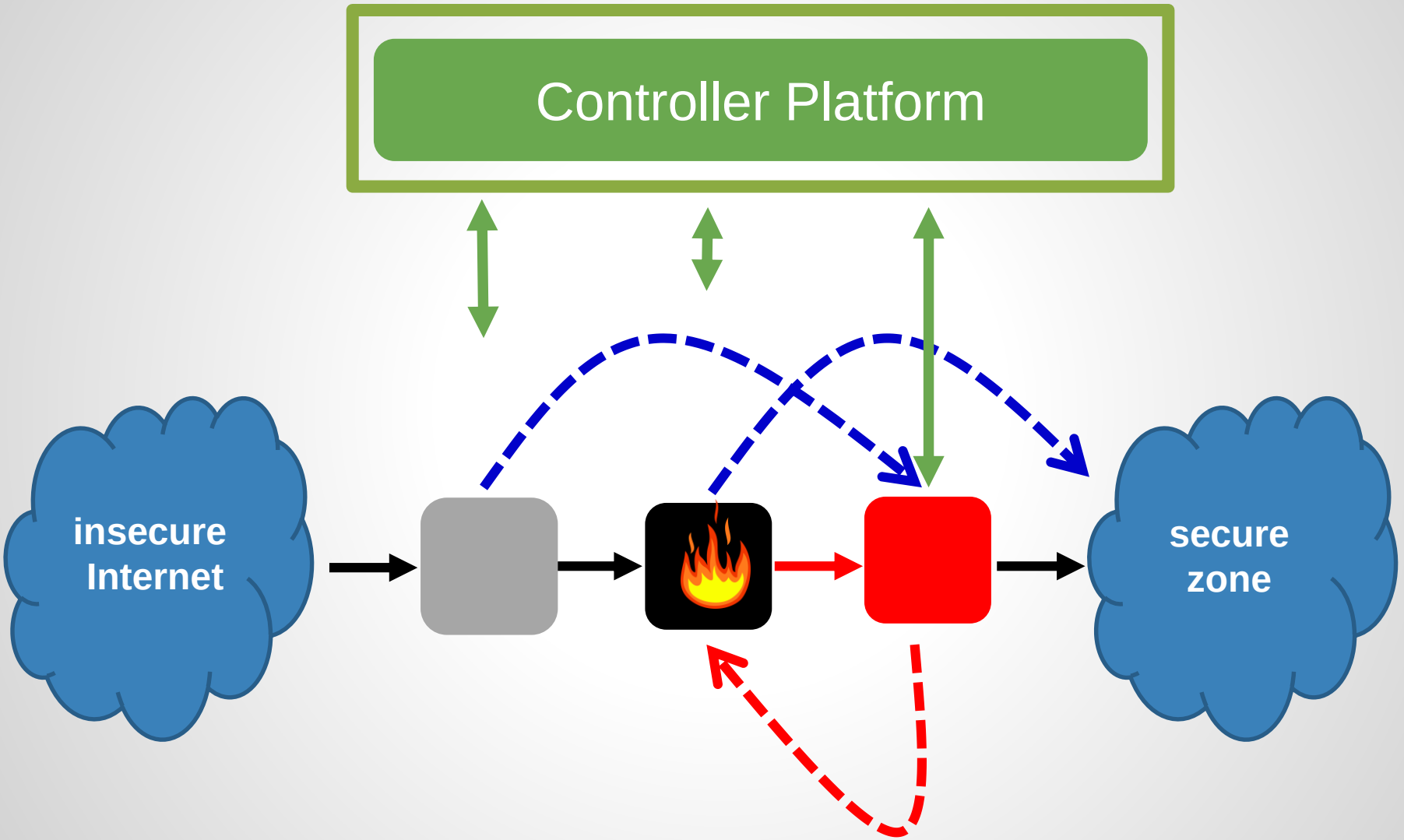
# What Can Go Wrong?



# Example 1: Bypassed Waypoint



# Example 2: *Transient* Loop

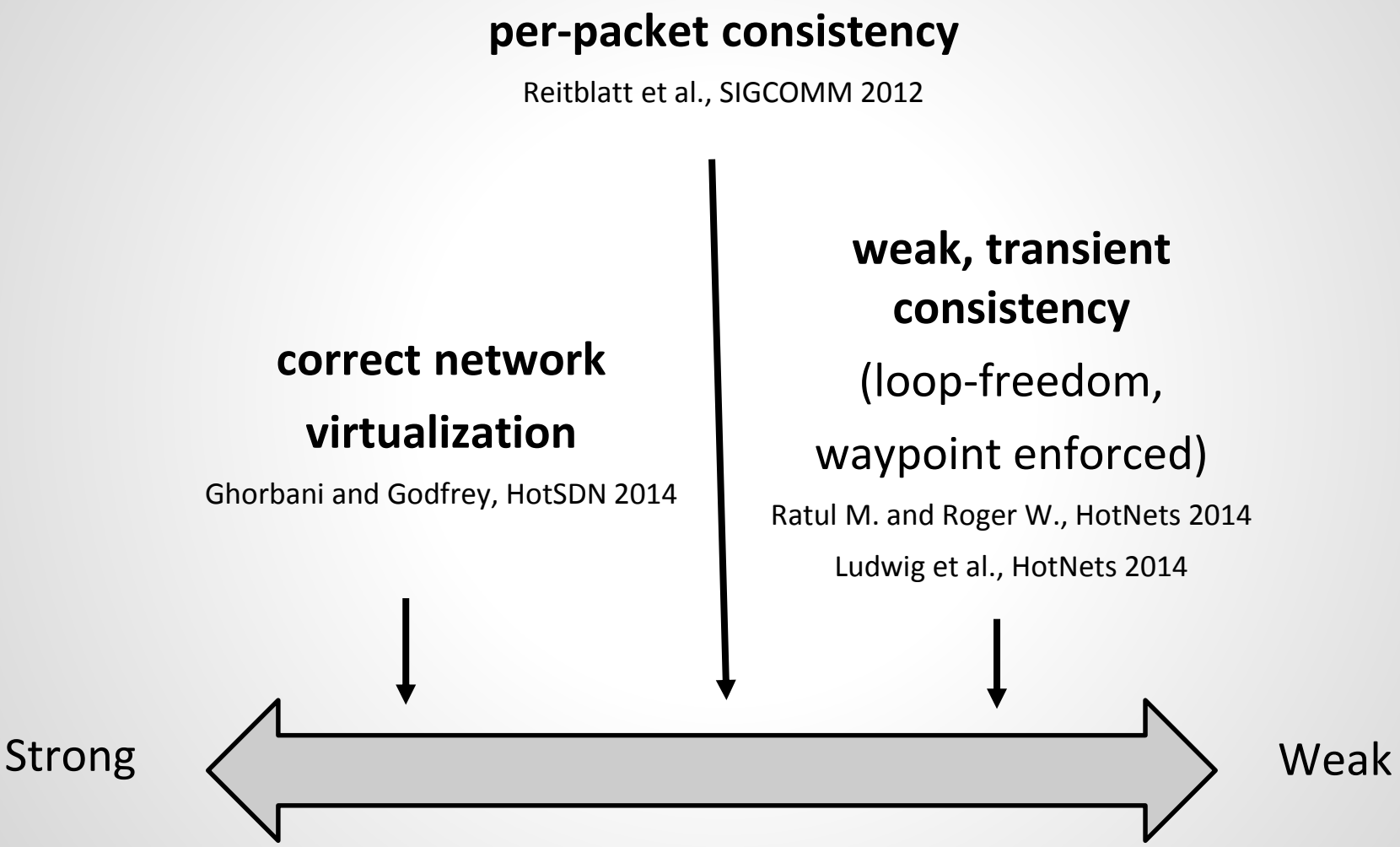


**What kind and level of consistency is needed?**

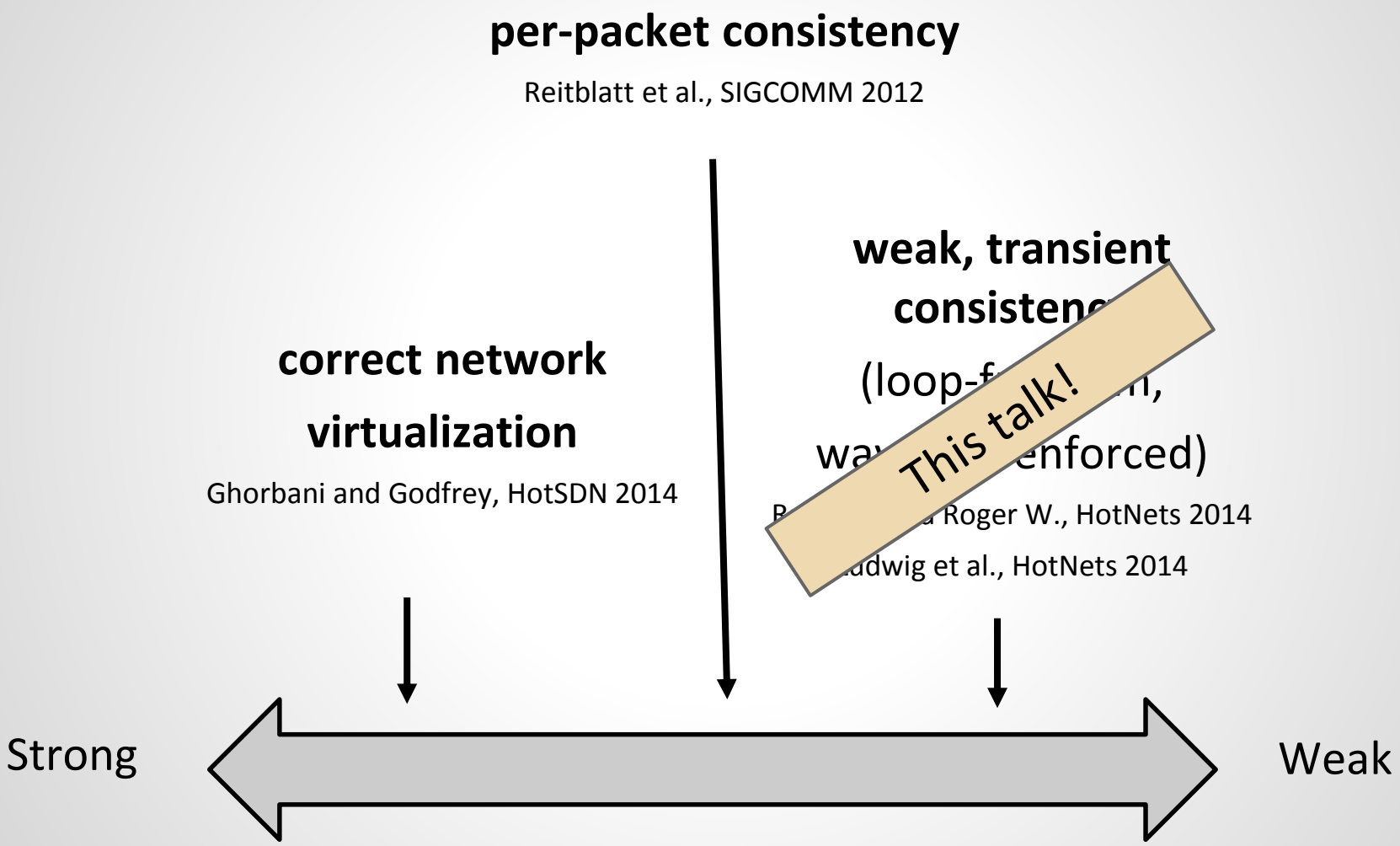
**What kind and level of consistency is needed?**

**It depends ☺**

# The Spectrum of Consistency

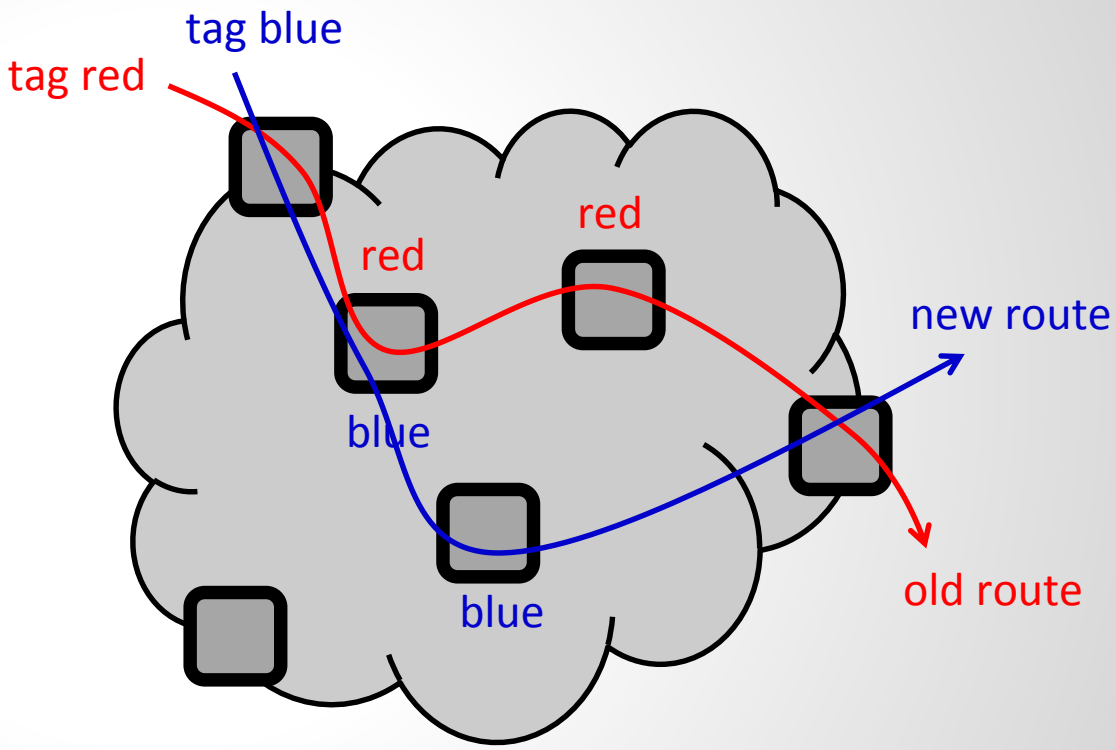


# The Spectrum of Consistency



# Almost everything can be solved with tagging…

- Old route: red
- New route: blue
- 2-Phase Update:
  - Install blue flow rules internally
  - Flip tag at ingress ports



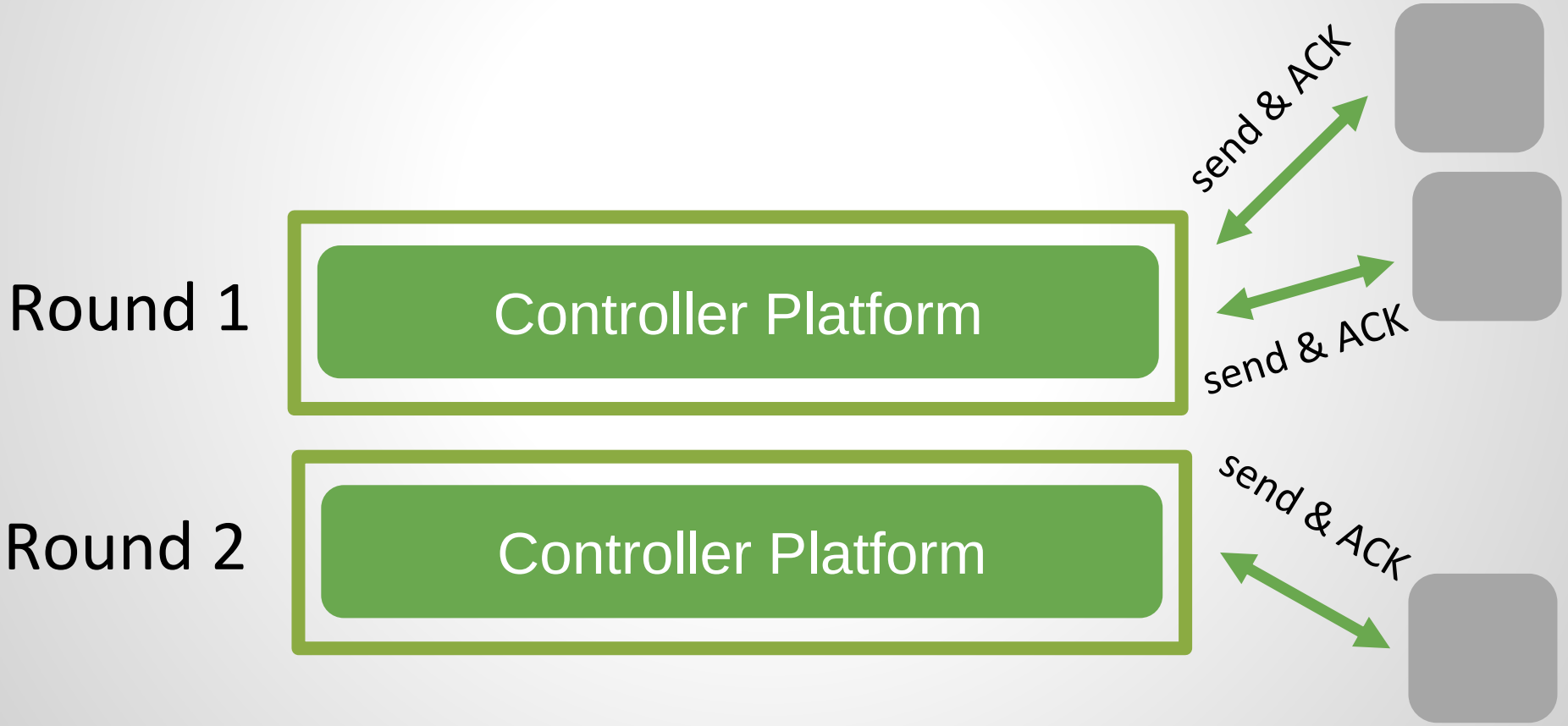
# The Case Against Tagging

- Correctness:
  - Where to tag? Don't interfere with existing protocols!
  - Tagging in the presence of middleboxes?
- Overhead:
  - Header space is limited
  - Looking up special header fields and tagging: extra latency?
  - The approach requires extra rules on the switch (TCAM memory is a scarce resource)
  - Coordination problem for distributed controllers?
- Late updates:
  - Updates start taking place late*

* Mahajan & Wattenhofer, ACM HotNets 2013

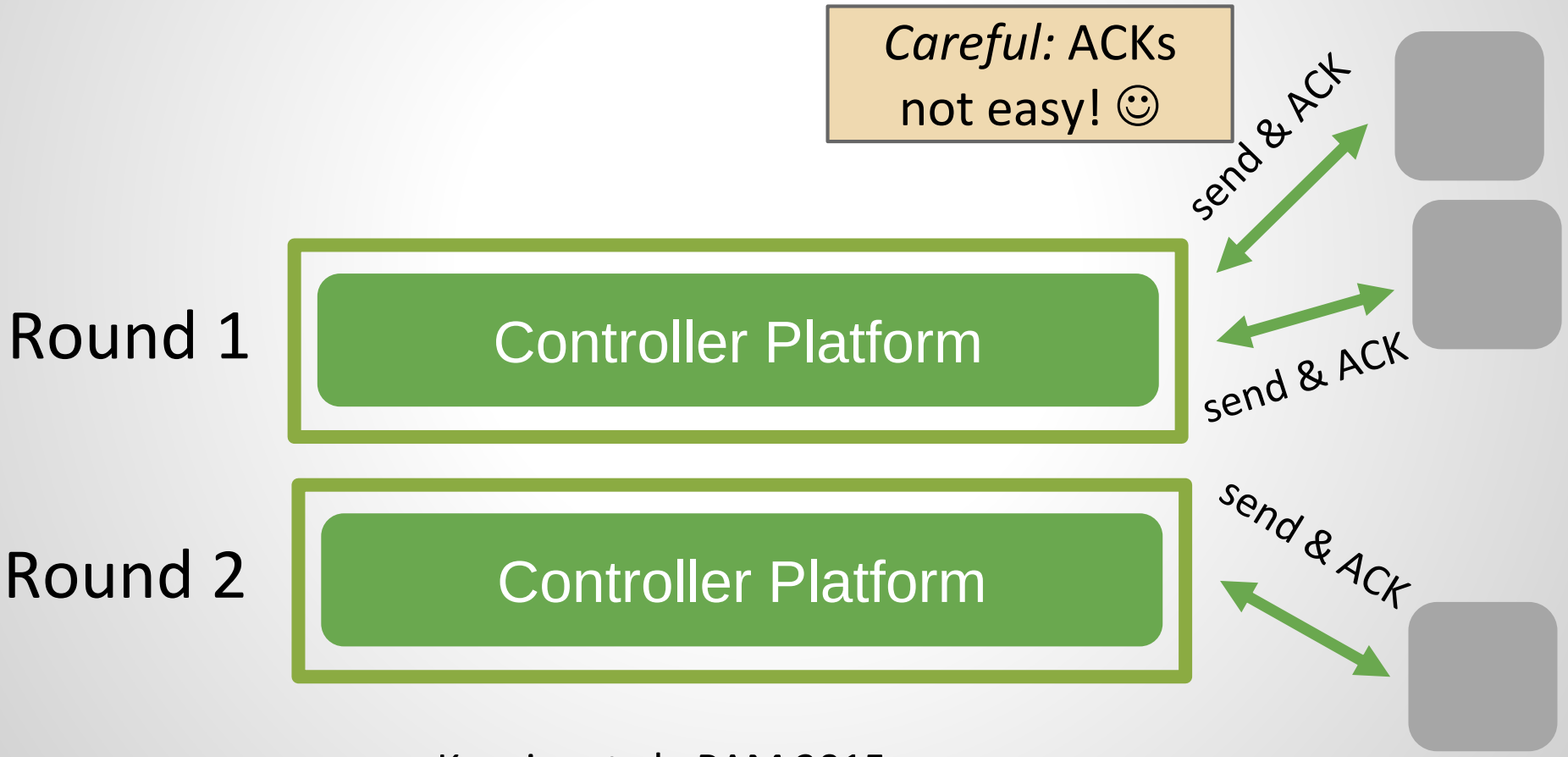
# Transient Consistency: Model

Idea: Keep consistent by updating in multiple rounds



# Transient Consistency: Model

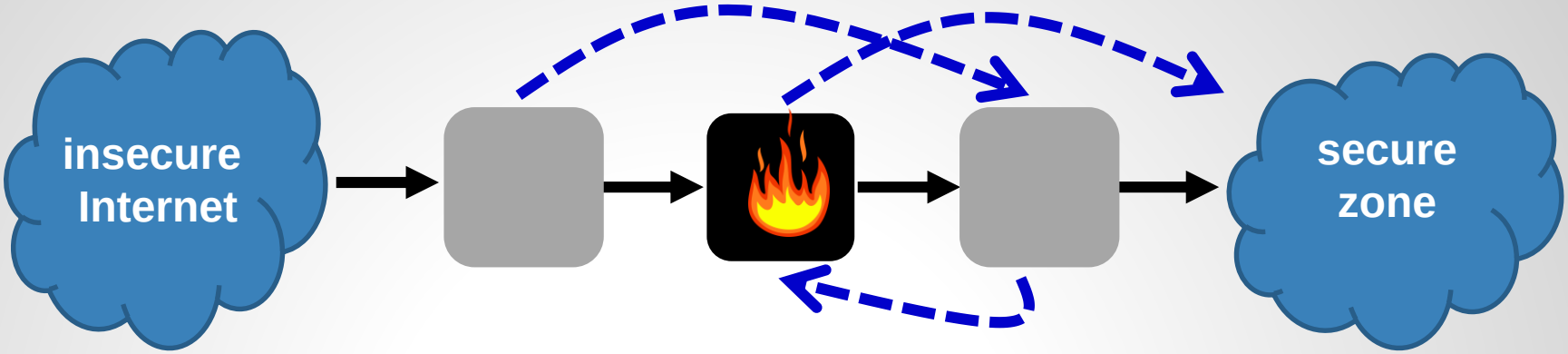
Idea: Keep consistent by updating in multiple rounds



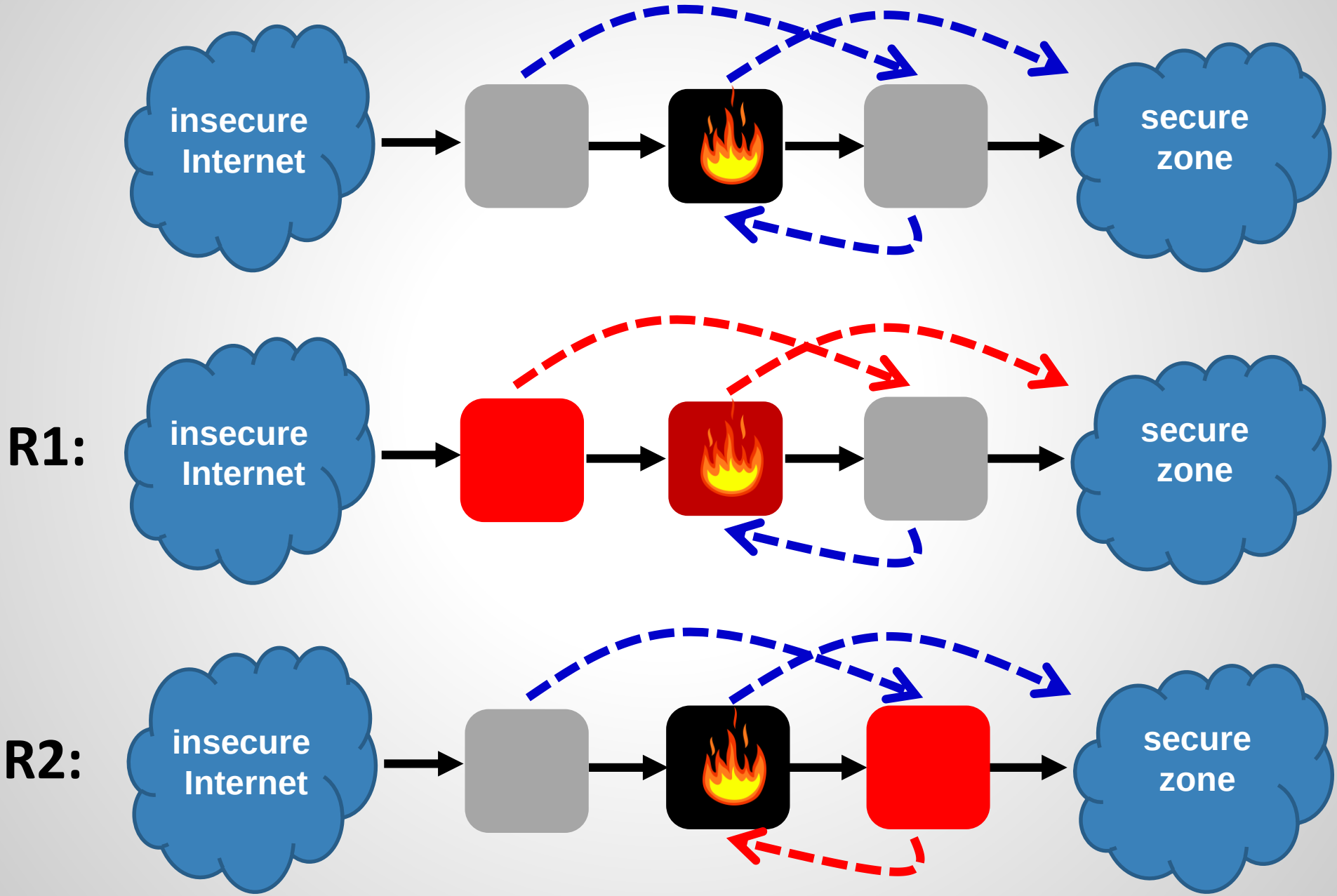

Kuzniar et al., PAM 2015

Kuzniar et al., ACM CONEXT 2014

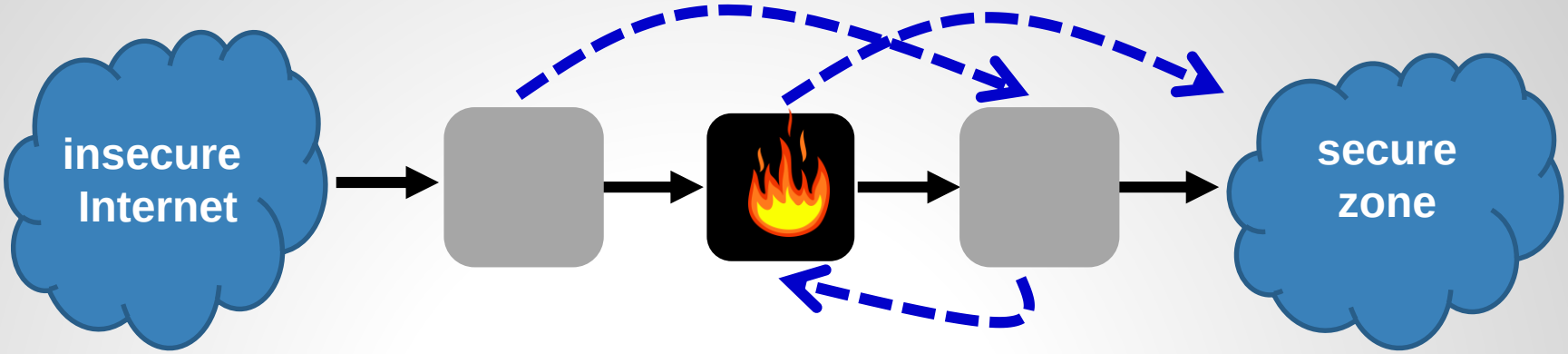
# Going Back to Our Examples: LF Update?



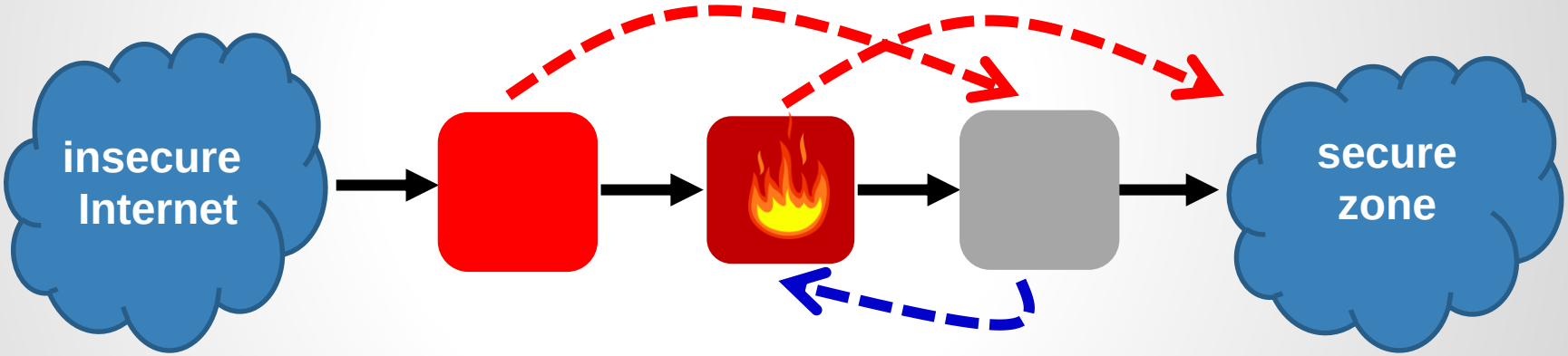
# Going Back to Our Examples: LF Update!



# Going Back to Our Examples: LF Update!




R1:

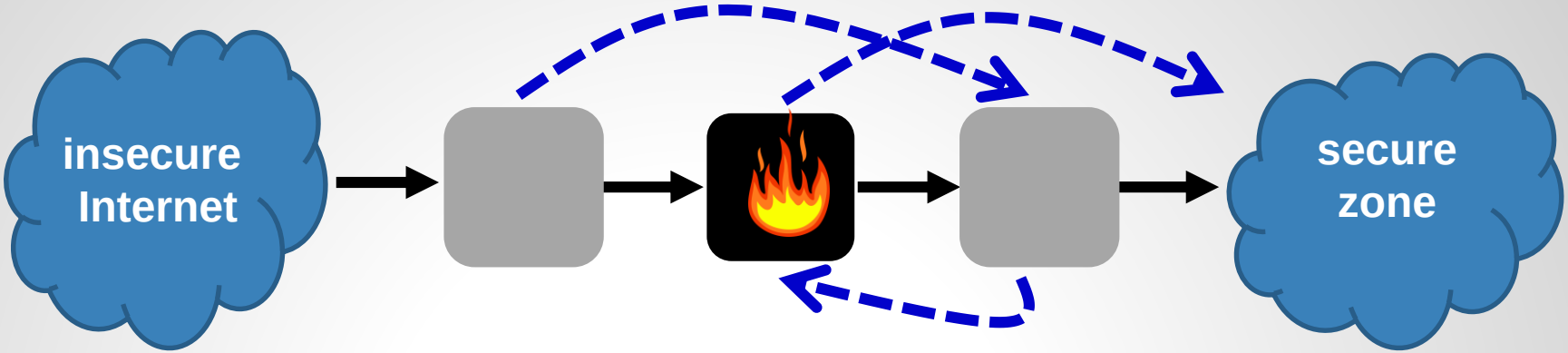



R2:

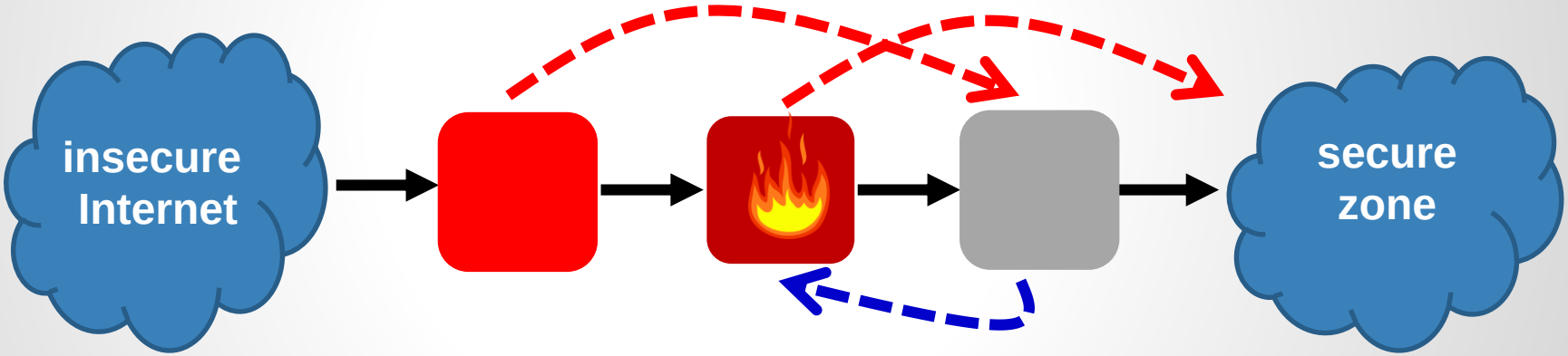
LF ok! But:

- Q1: Does a LF schedule always exist? Ideas?

# Going Back to Our Examples: LF Update!




R1:

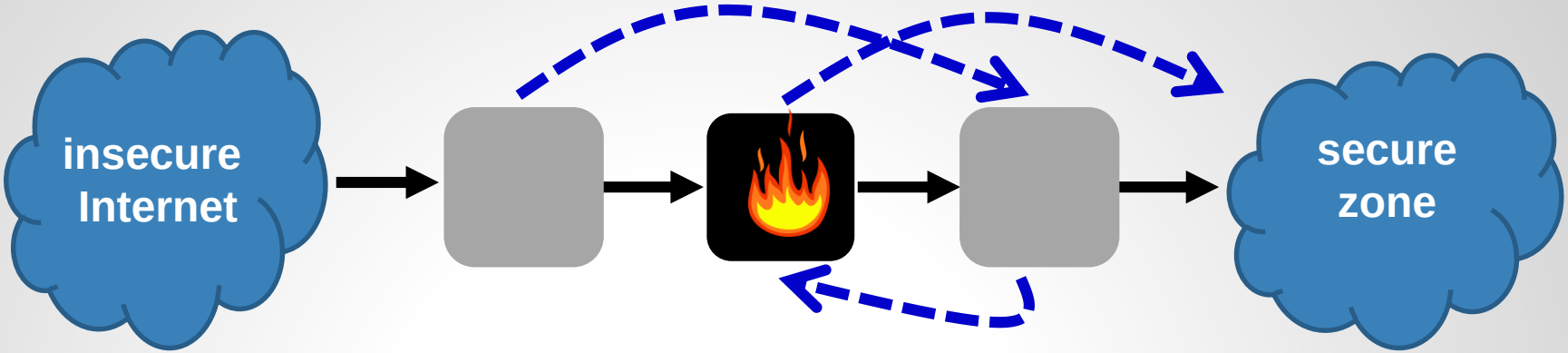



R2:

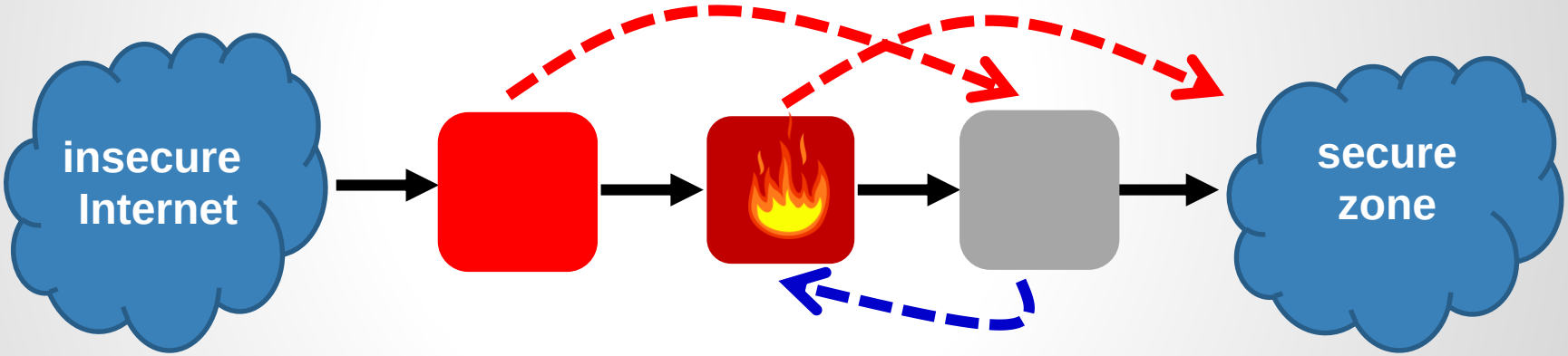
LF ok! But:

- Q1: Does a LF schedule always exist? Ideas?
- Q2: What about WPE?

# Going Back to Our Examples: LF Update!




R1:

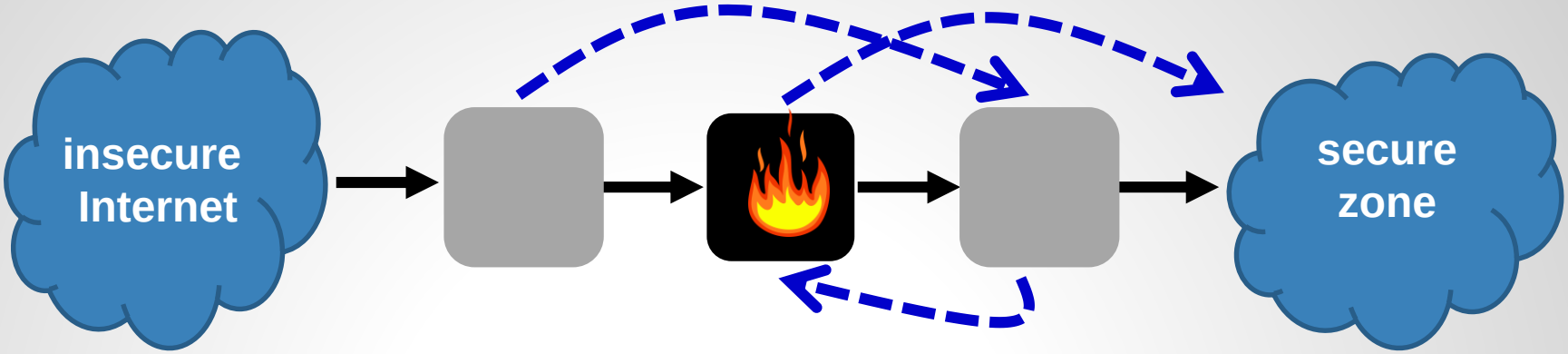



R2:

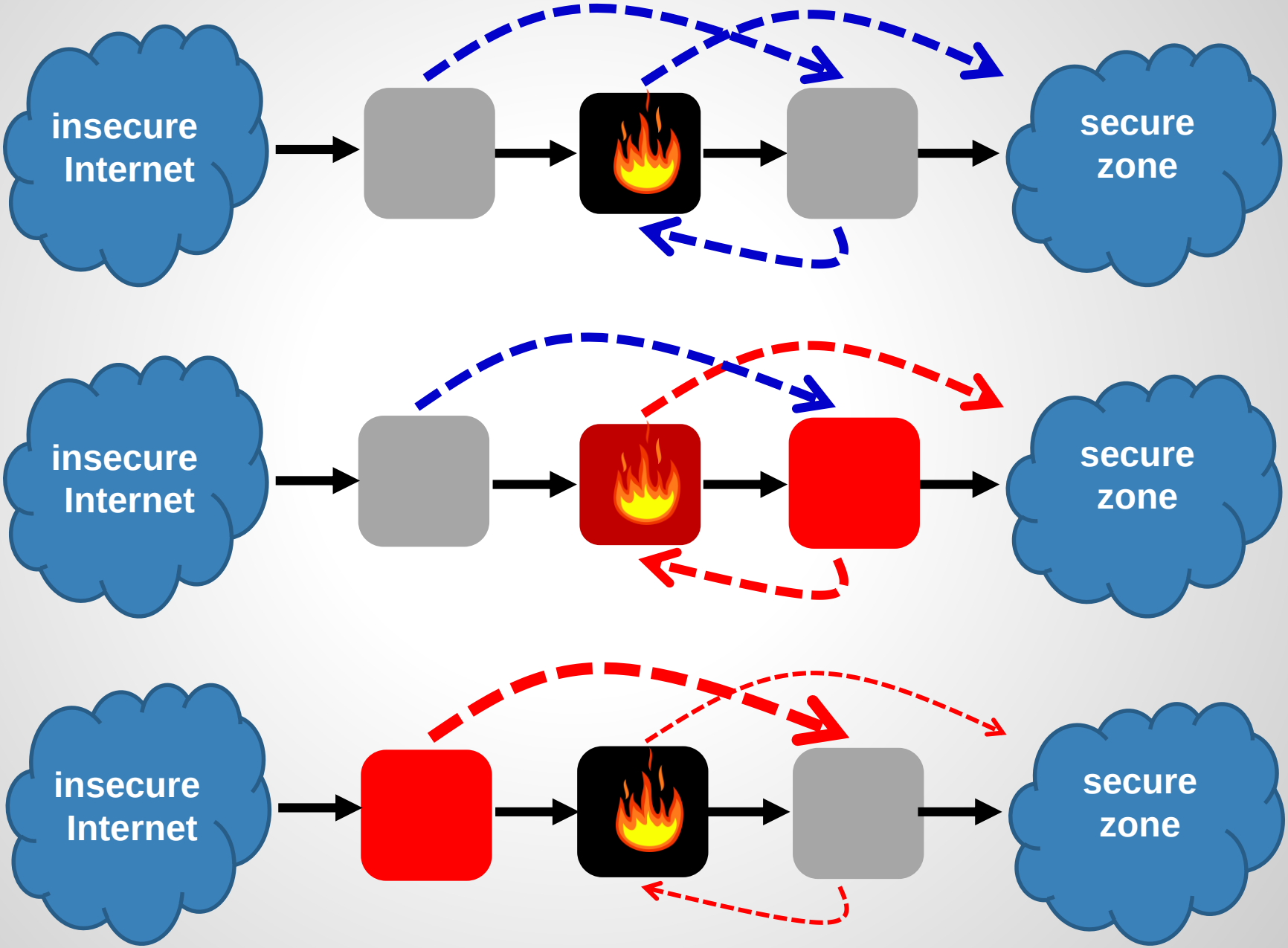
LF ok! But:

- Q1: Does a LF schedule always exist? Ideas?
- Q2: What about WPE? Violated in Round 1!

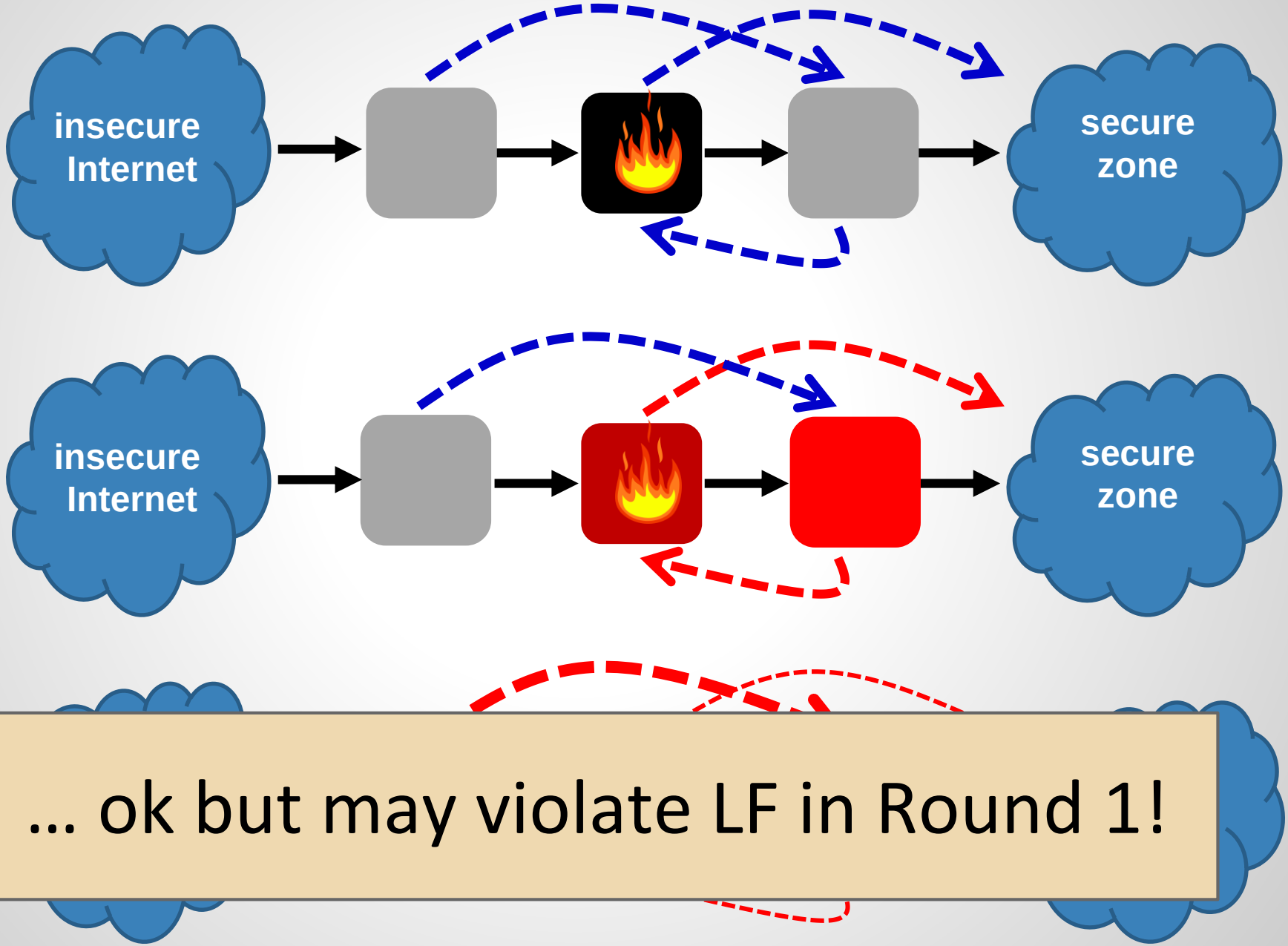
# Going Back to Our Examples: WPE Update?



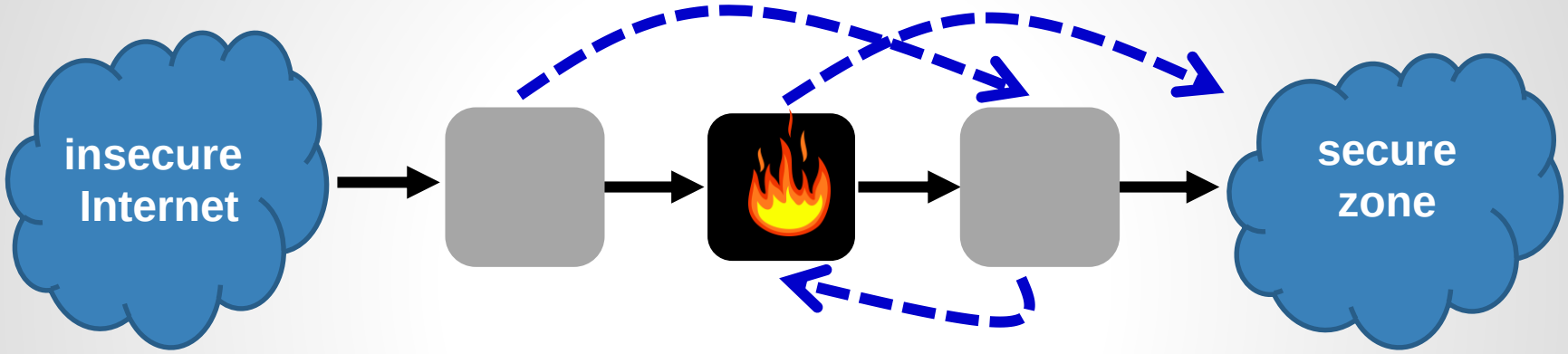
# Going Back to Our Examples: WPE Update!



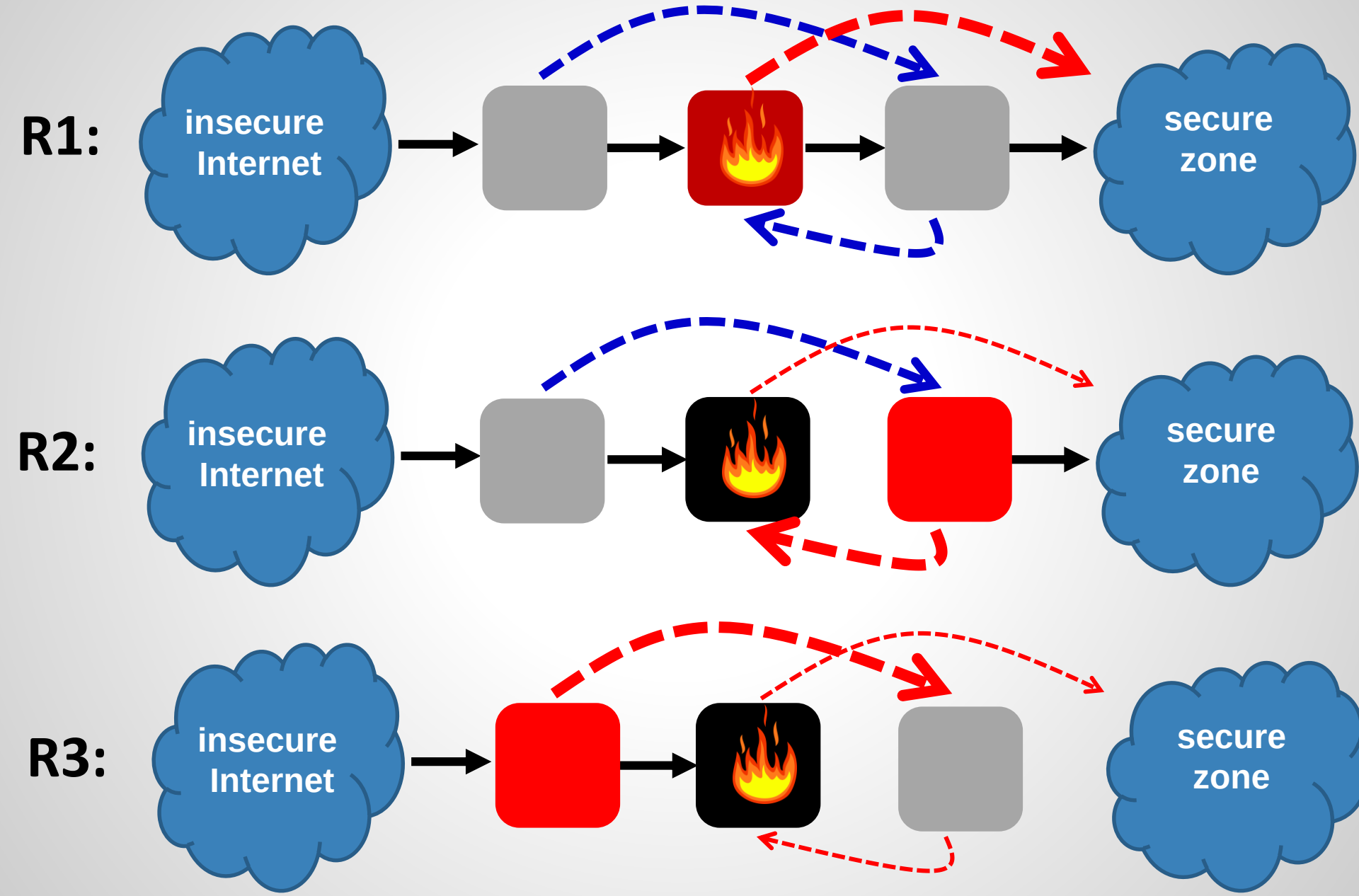
# Going Back to Our Examples: WPE Update!



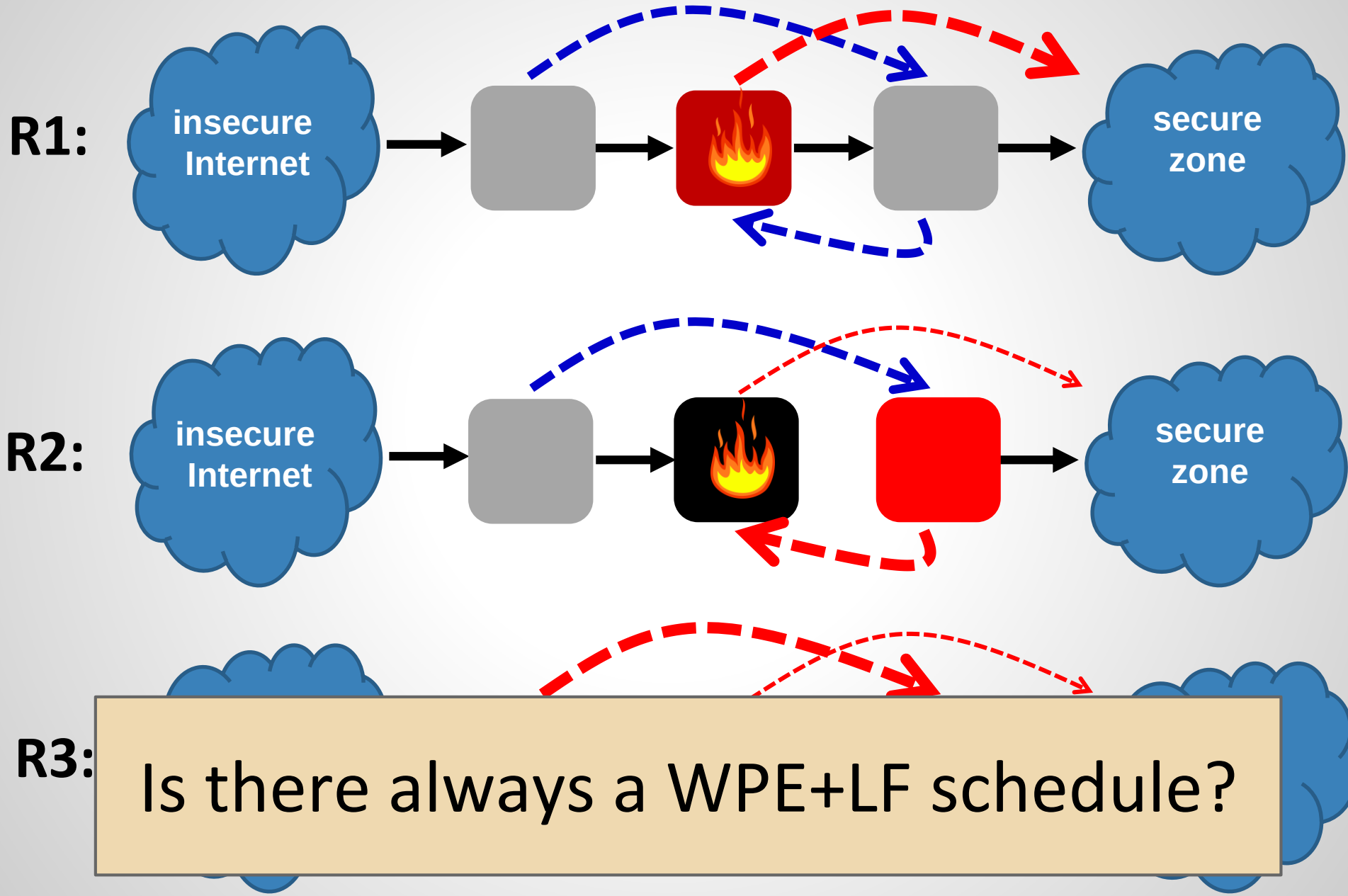
# Going Back to Our Examples: Both WPE+LF?



# Going Back to Our Examples: WPE+LF!

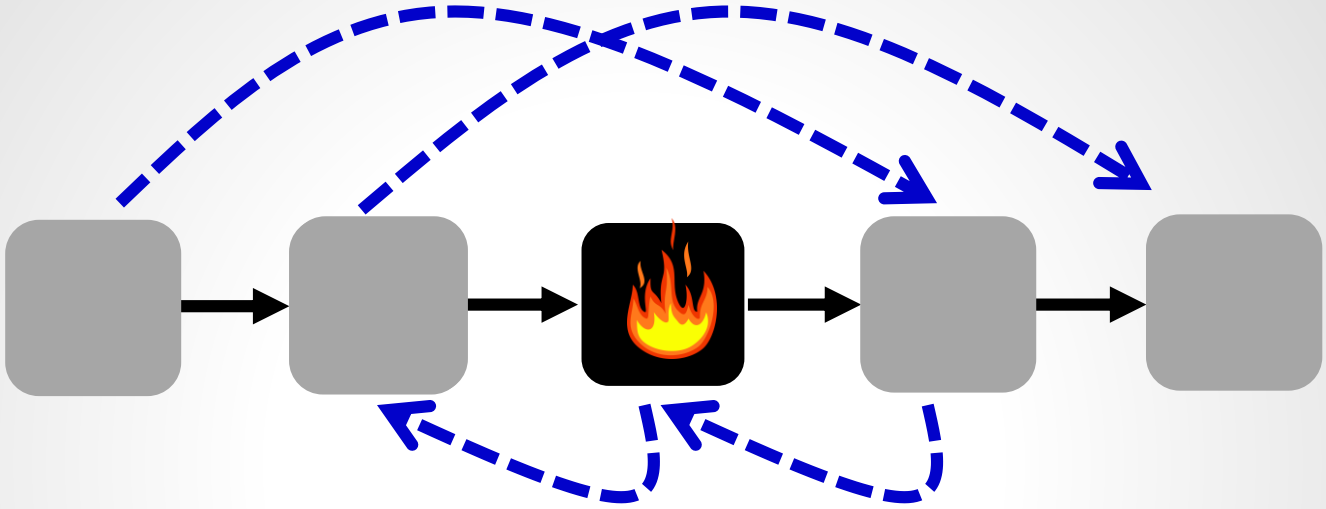


# Going Back to Our Examples: WPE+LF!

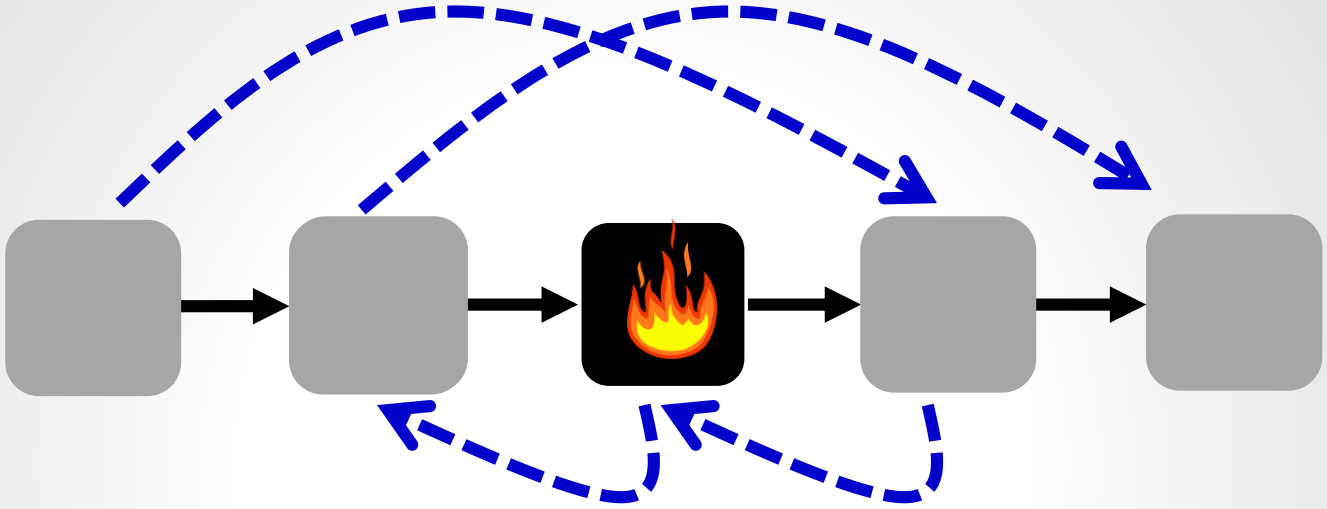



Is there always a WPE+LF schedule?

# What about this one?



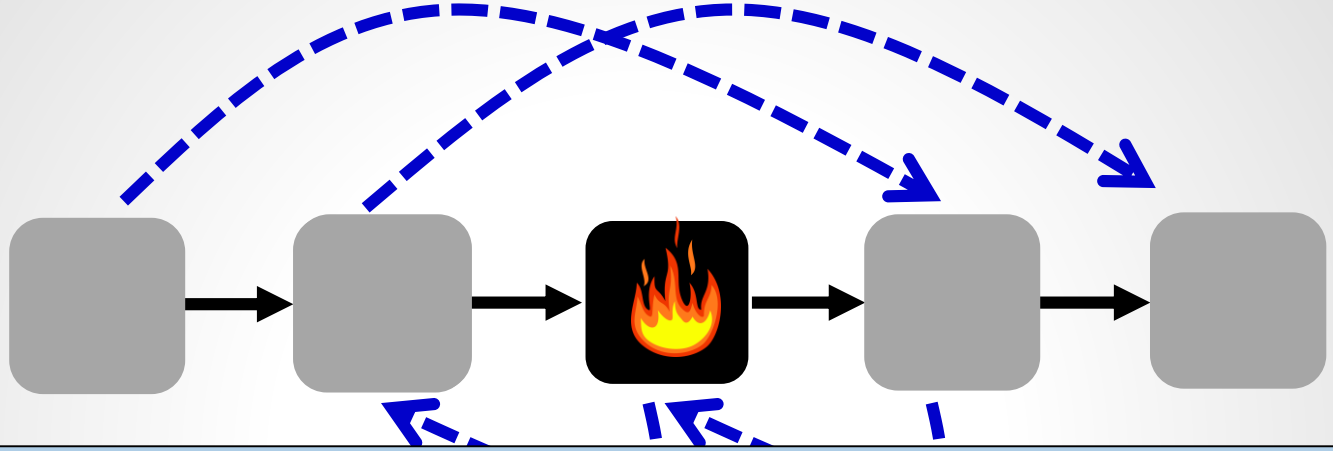
# LF and WPE may conflict!



- ❑ Cannot update any forward edge in R1: WP
- ❑ Cannot update any backward edge in R1: LF

No schedule exists!

# LF and WPE may conflict!



Can't Touch This: Consistent Network Updates for Multiple Policies
Szymon Dudycz, Arne Ludwig, and Stefan Schmid.
46th IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**), Toulouse, France, June 2016.

- ❑ Cannot update anv

Transiently Secure Network Updates
Arne Ludwig, Szymon Dudycz, Matthias Rost, and Stefan Schmid.
42nd ACM **SIGMETRICS**, Antibes Juan-les-Pins, France, June 2016.

- ❑ Ca

Scheduling Loop-free Network Updates: It's Good to Relax!
Arne Ludwig, Jan Marcinkowski, and Stefan Schmid.
ACM Symposium on Principles of Distributed Computing (**PODC**), Donostia-San Sebastian, Spain, July 2015.

Good Network Updates for Bad Packets: Waypoint Enforcement Beyond Destination-Based Routing Policies
Arne Ludwig, Matthias Rost, Damien Foucard, and Stefan Schmid.
13th ACM Workshop on Hot Topics in Networks (**HotNets**), Los Angeles, California, USA, October 2014.

# Challenges of More Flexible Networked Systems

1. Kraken: Predictable cloud application performance through adaptive virtual clusters

2. C3: Low tail latency in cloud data stores through replica selection

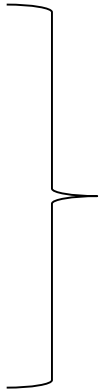
3. Peacock: Consistent network updates

4. **Panopticon: How to introduce these innovative technologies in the first place? Case study: SDN**

# SDN Use Cases Today

Many use cases discussed today, e.g. in:

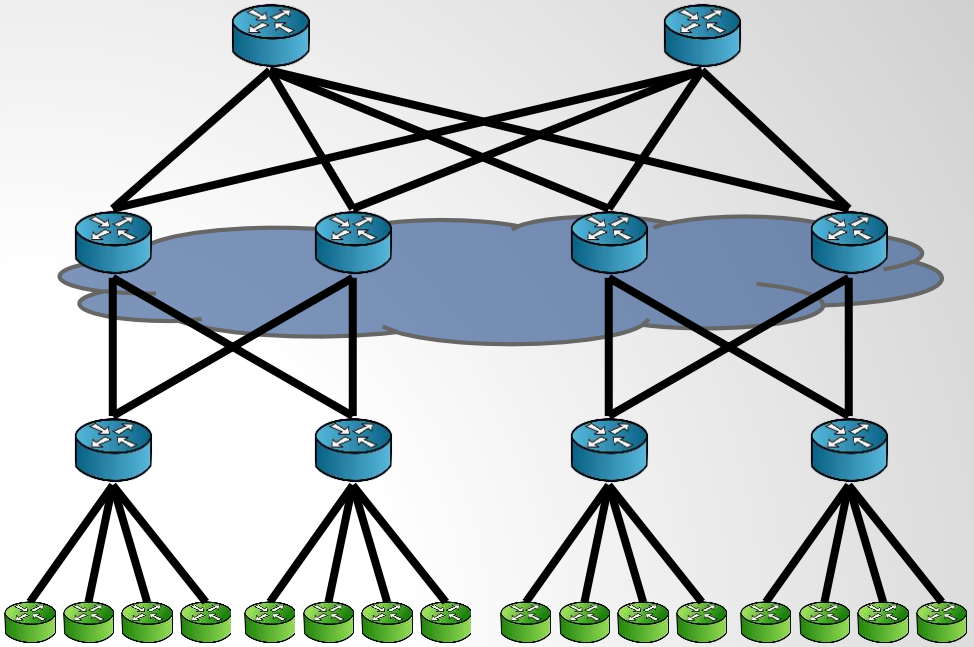
- Enterprise networks
- Datacenters
- WANs
- IXPs
- ISPs



Existing deployments!

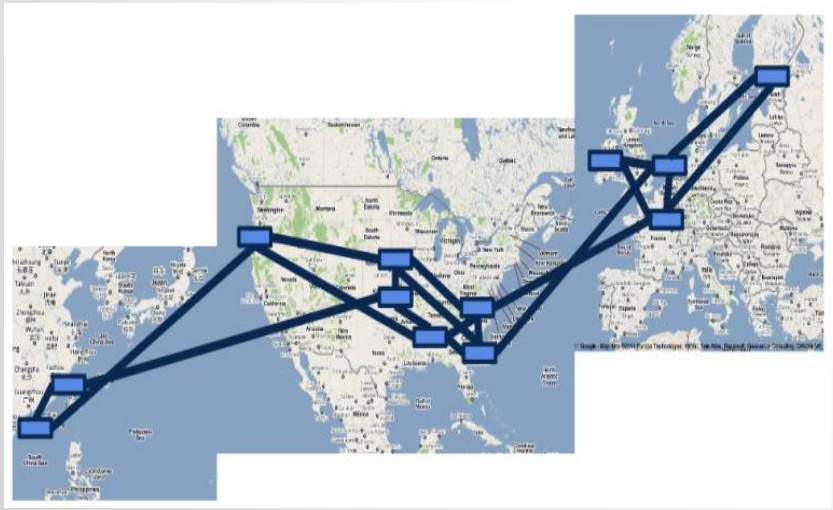
**How to deploy SDN cost effectively?**

# SDN Deployment



Datacenter: Easy

- SDN can be deployed at **software edge** (terminate links at Open vSwitch)
- 2 Control Planes: **ECMP Fabric**



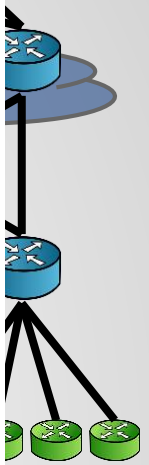
WAN: «Easy»

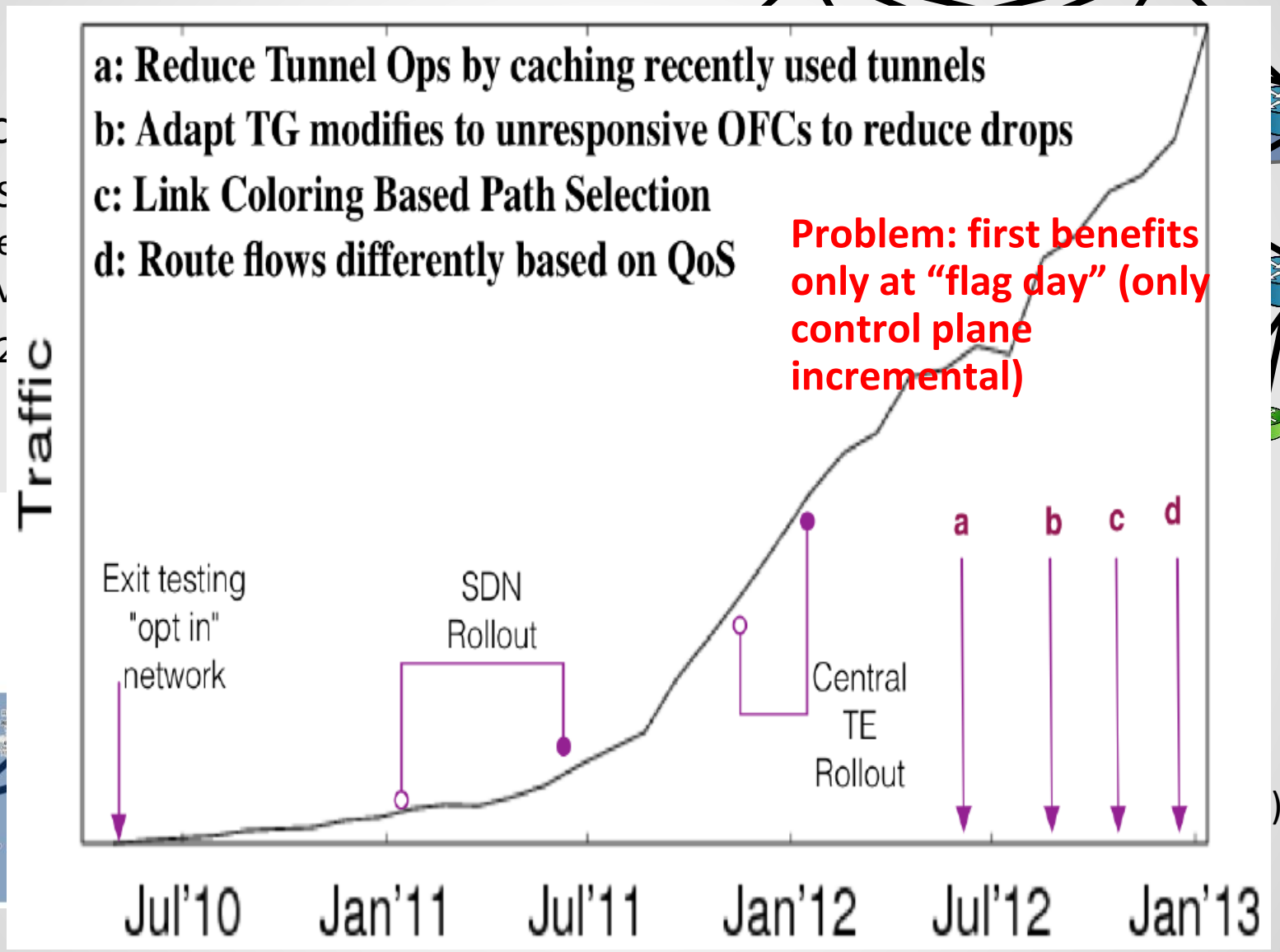
- Google B4: **small network**
- Can be deployed at end of long-haul fiber (replace IP core router)

# SDN Deployment

Datac

- S
  e
  v
- 2

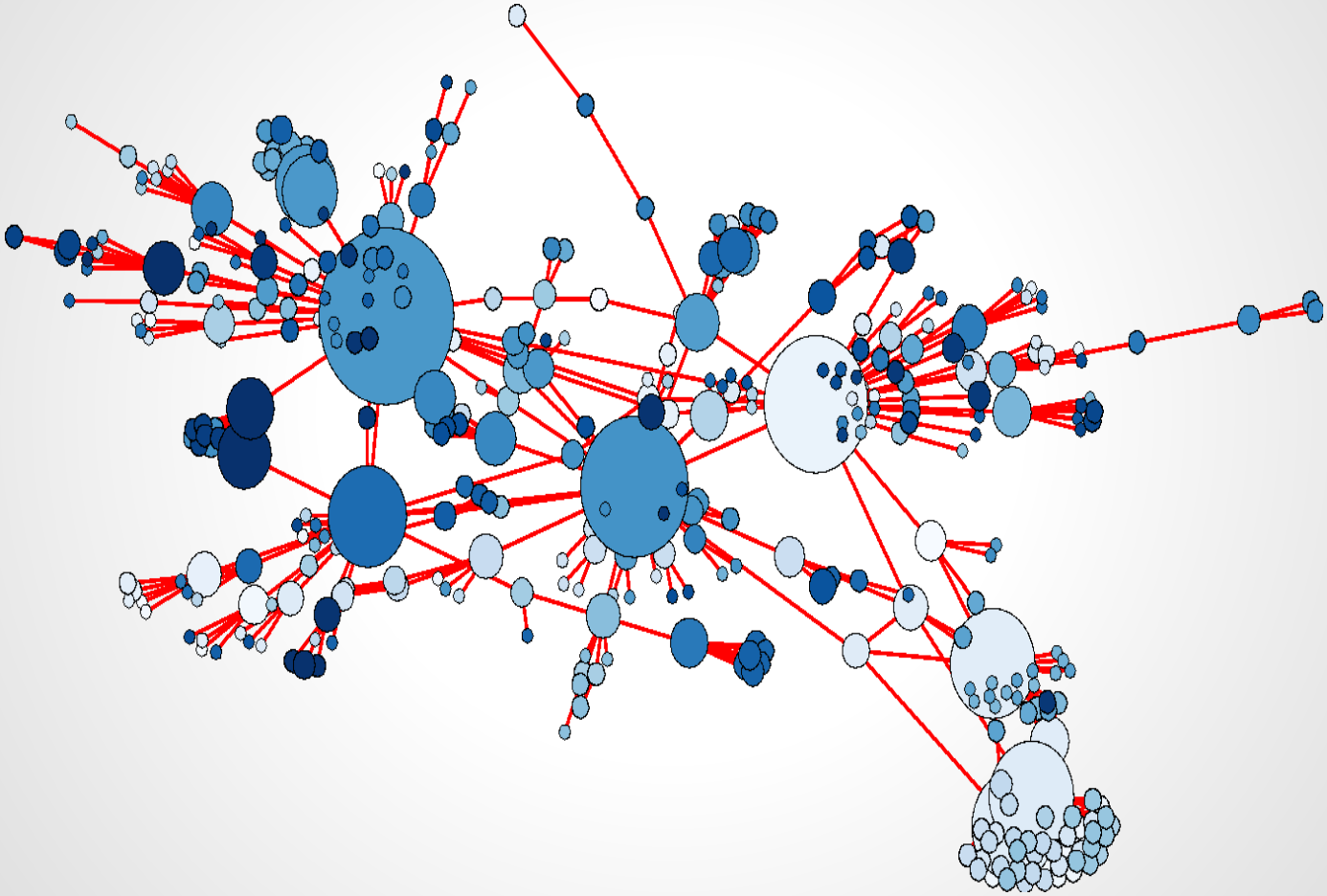




# But how to deploy SDN in enterprise?

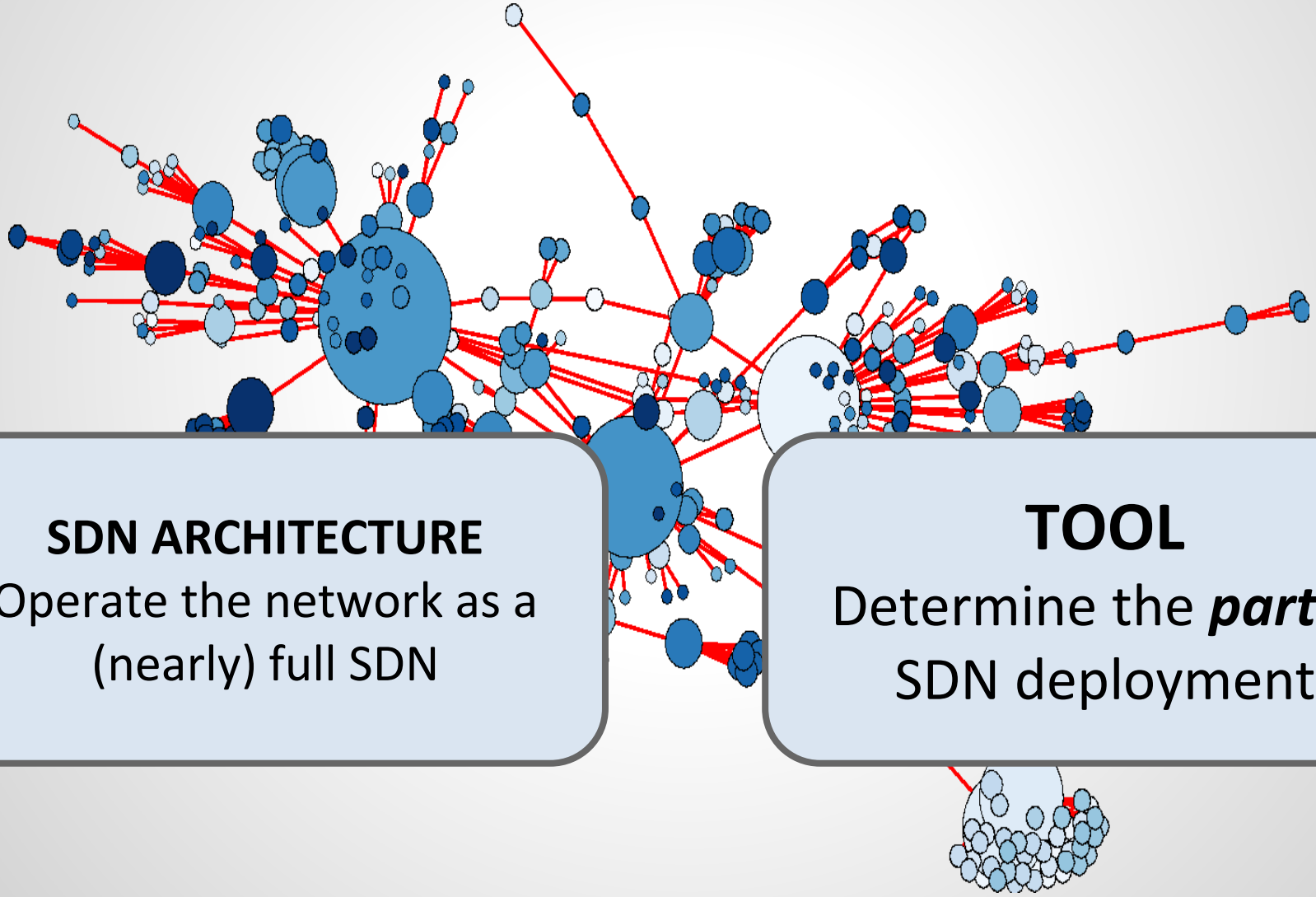
- Large and complex networks, **budgets limited**
- Idea: Can we **incrementally deploy SDN** into enterprise campus networks?
- And what **SDN benefits** can be realized in a hybrid deployment?

# Can we deploy SDN at enterprise edge?



**The edge is large, and not in software!**

# Panopticon

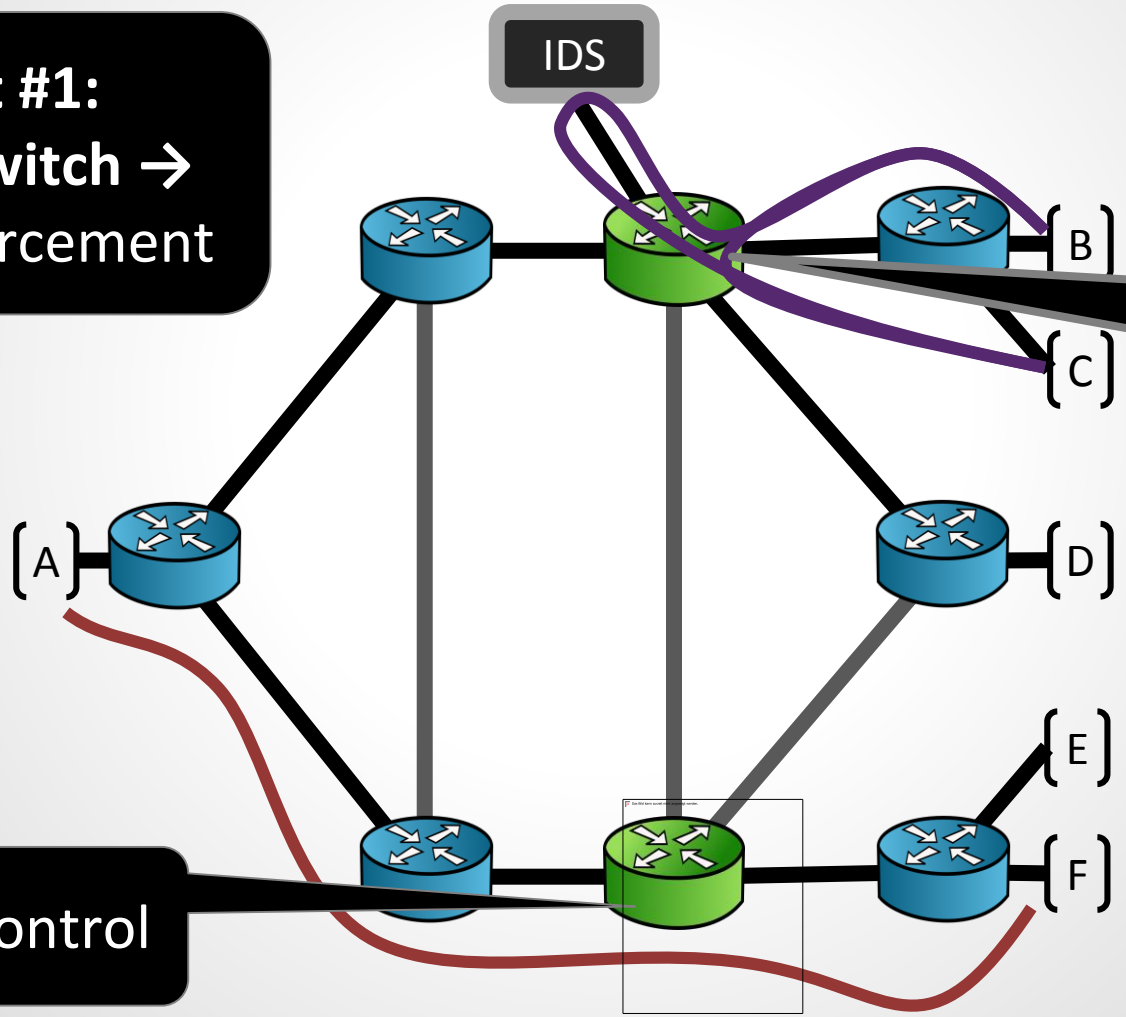


**SDN ARCHITECTURE**
Operate the network as a (nearly) full SDN

**TOOL**
Determine the ***partial*** SDN deployment

# Get Functionality with Waypoint Enforcement

**Insight #1:**
**≥ 1 SDN switch →**
Policy enforcement

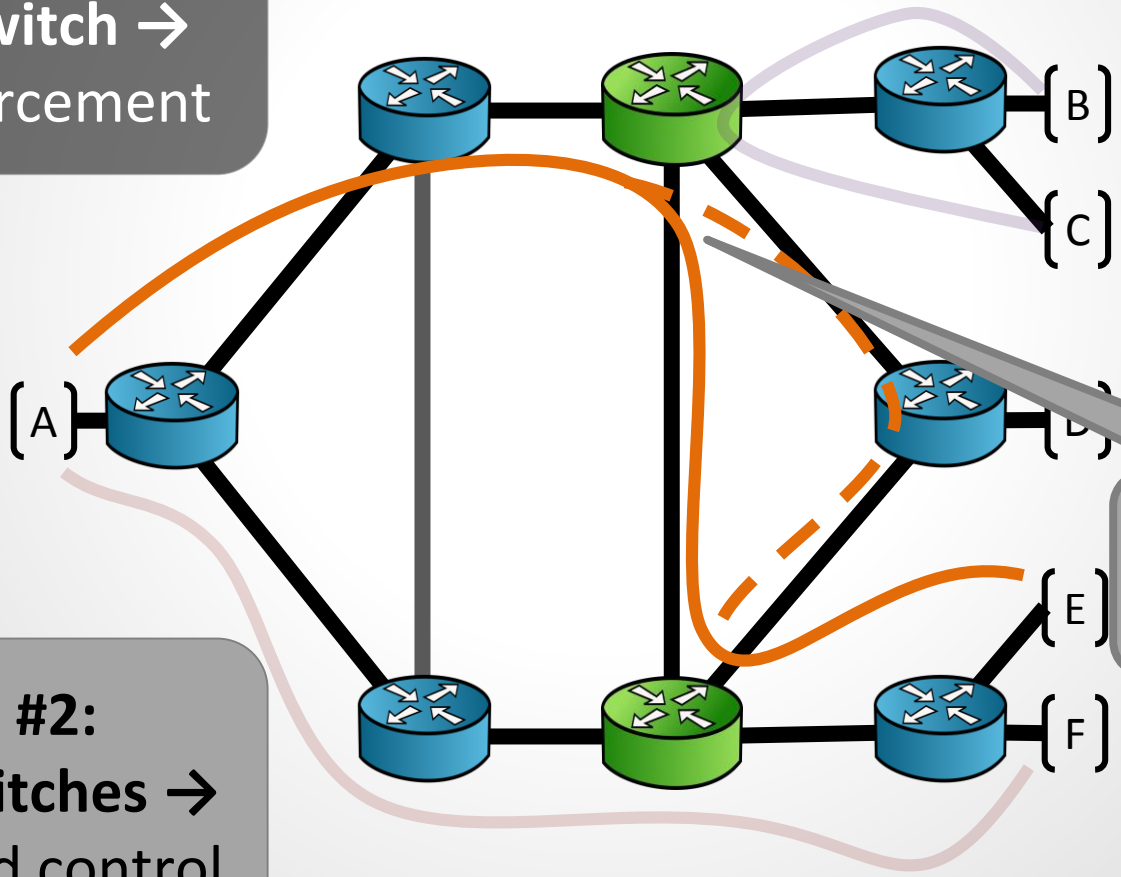



Middlebox traversal

Access control

# Larger Deployment = More Flexibility

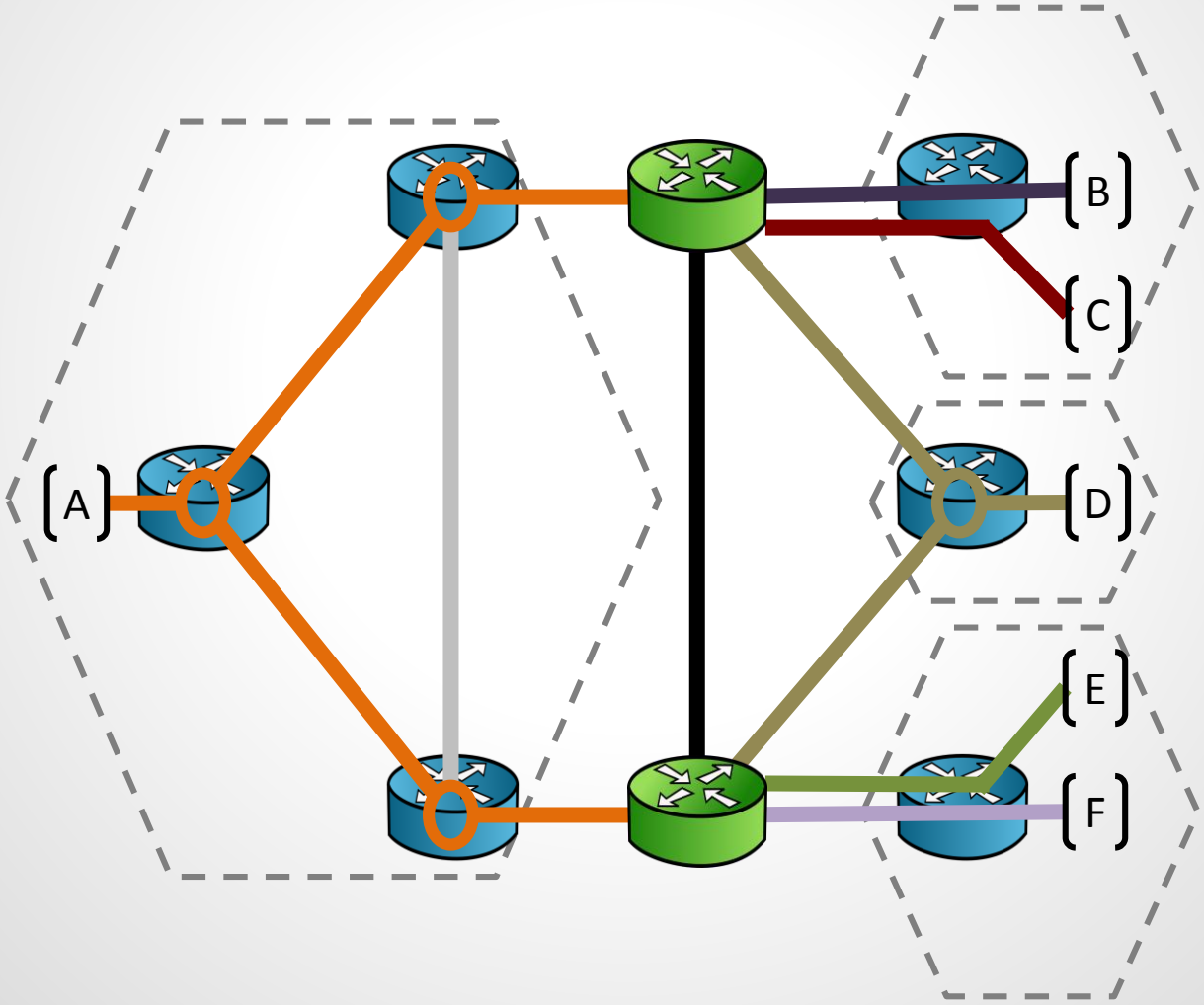
**Insight #1:**
**≥ 1 SDN switch →**
Policy enforcement




**Insight #2:**
**≥ 2 SDN switches →**
Fine-grained control

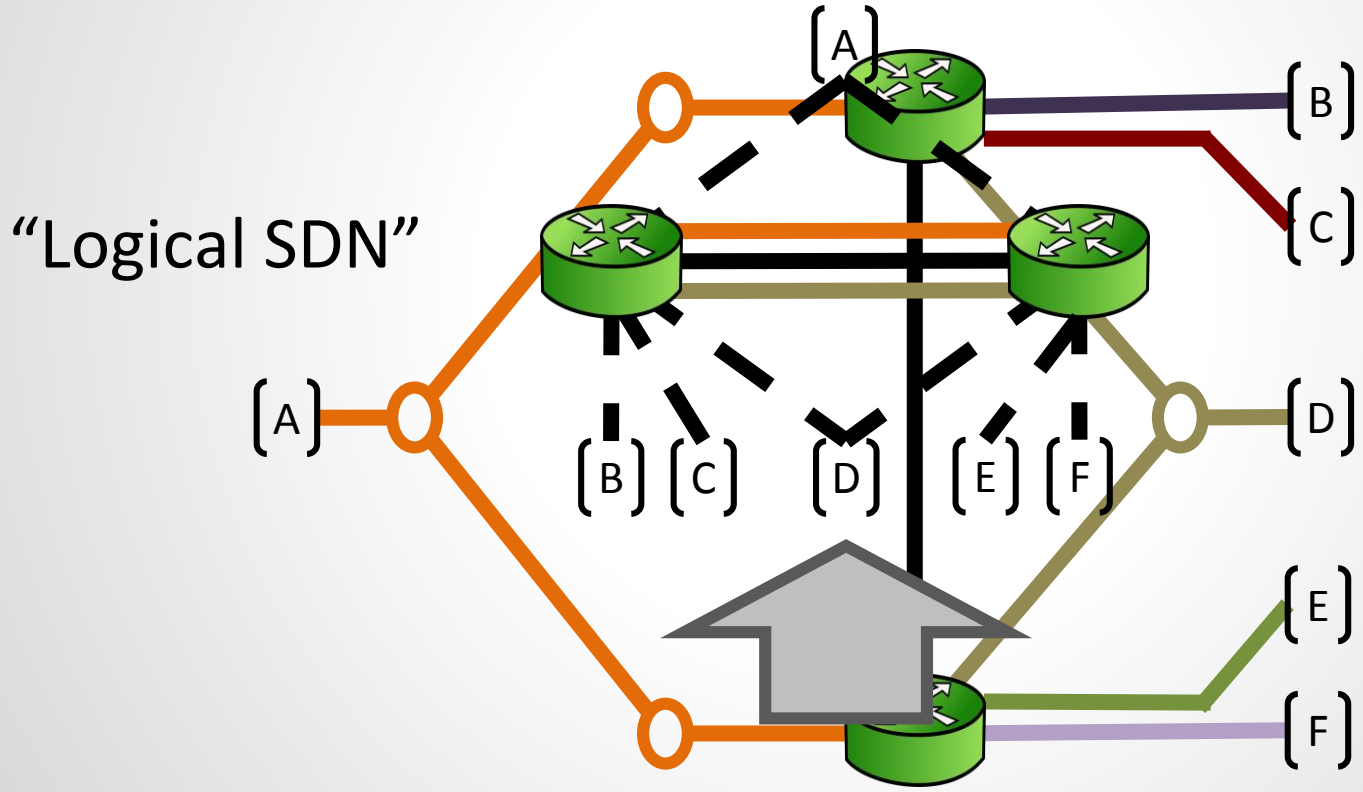
# Panopticon: Building the Logical SDN Abstraction

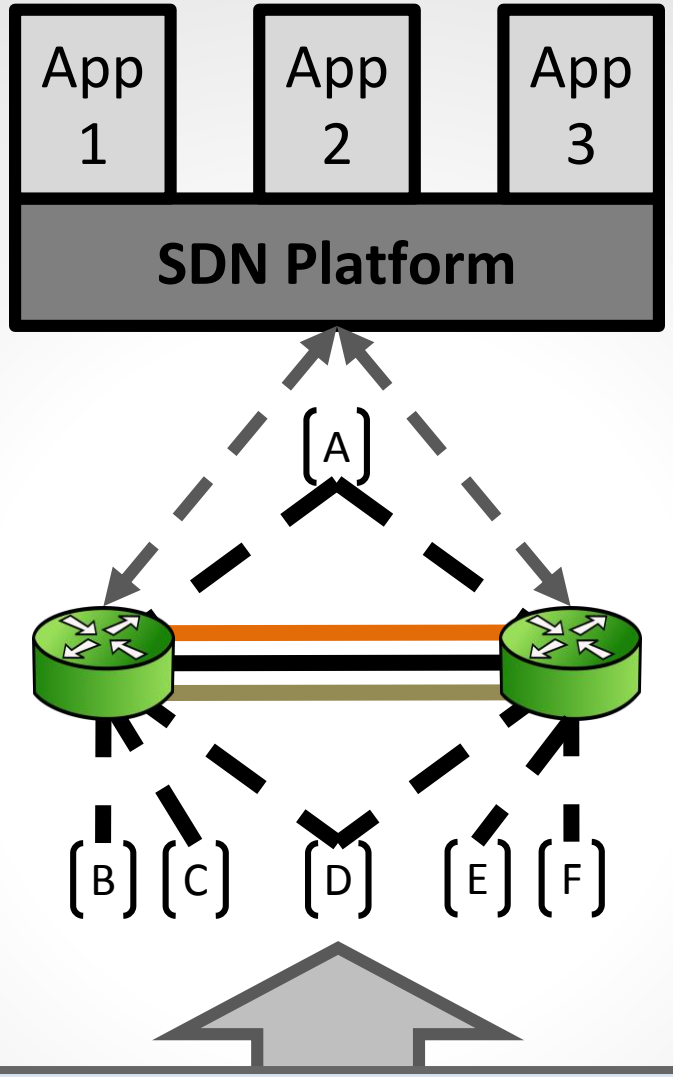
1. Restrict traffic by using VLANs



# Panopticon: Building the Logical SDN Abstraction

2. Build logical SDN

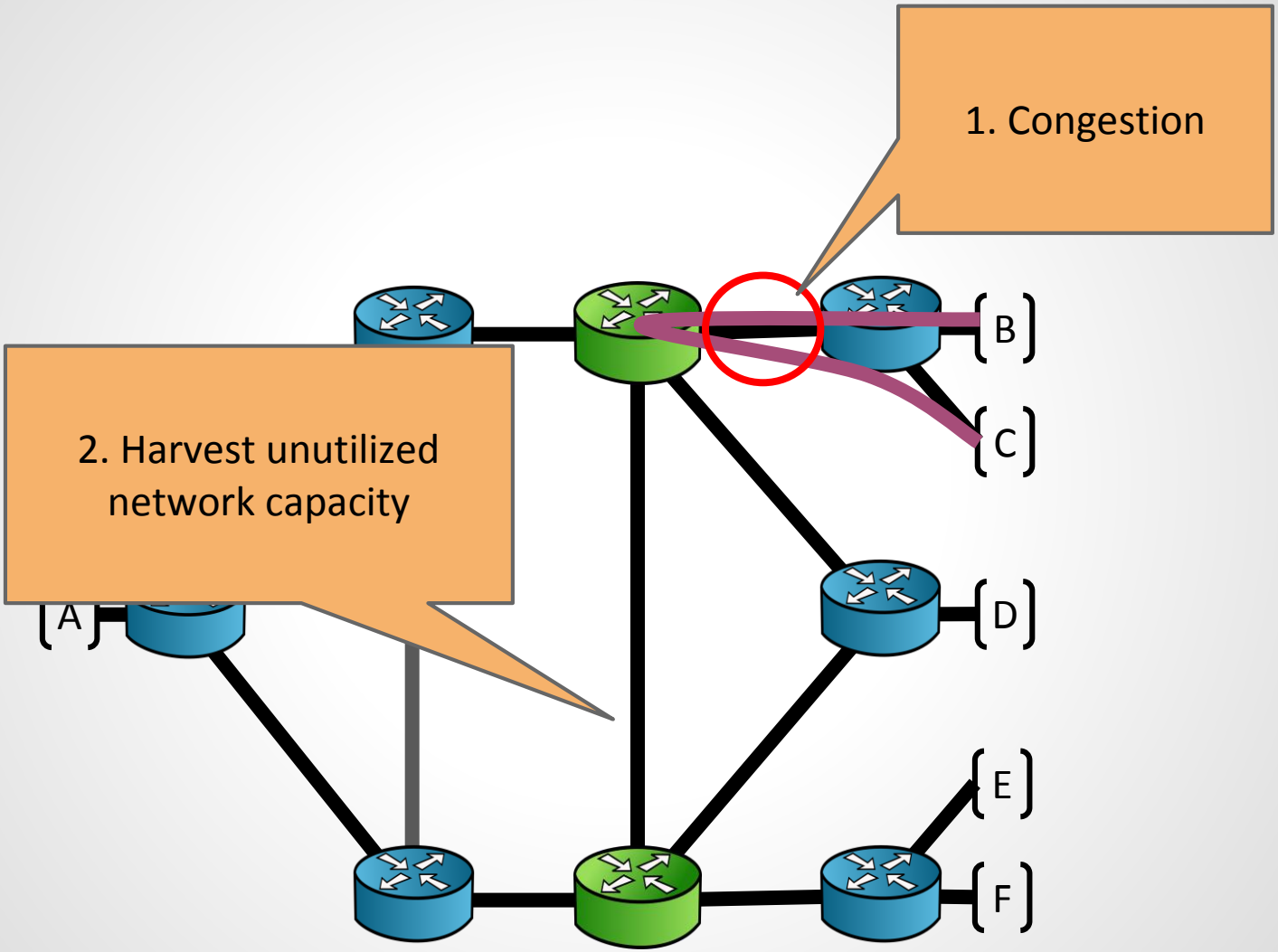




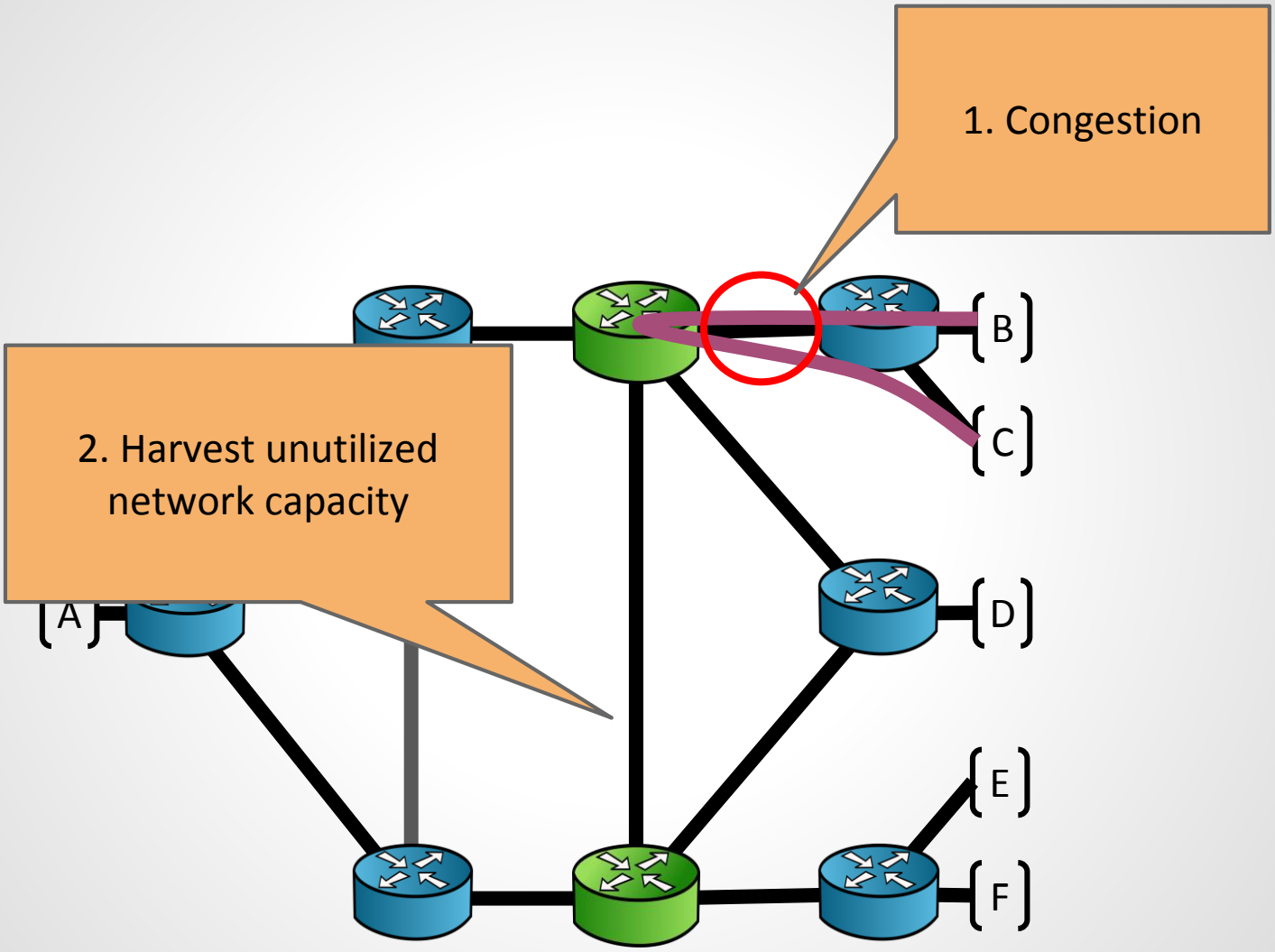

"Logical SDN"

**PANOPTICON provides the abstraction of a (nearly) fully-deployed SDN in a partially upgraded network**

# Good or Bad Impact on Traffic?



# Good or Bad Impact on Traffic?




Panopticon: Reaping the Benefits of Incremental SDN Deployment in Enterprise Networks
Dan Levin, Marco Canini, Stefan Schmid, Fabian Schaffert, and Anja Feldmann.
USENIX Annual Technical Conference (**ATC**), Philadelphia, Pennsylvania, USA, June 2014.

# Additional Dimensions

## ❑ NFV Placement

It's a Match! Near-Optimal and Incremental Middlebox Deployment
Tamás Lukovszki, Matthias Rost, and Stefan Schmid.
ACM SIGCOMM Computer Communication Review (**CCR**), January 2016.

Online Admission Control and Embedding of Service Chains
Tamás Lukovszki and Stefan Schmid.
22nd International Colloquium on Structural Information and Communication Complexity (**SIROCCO**), Montserrat, Spain, July 2015.

## ❑ Self-Adjusting Topologies

SplayNet: Towards Locally Self-Adjusting Networks
Stefan Schmid, Chen Avin, Christian Scheideler, Michael Borokhovich, Bernhard Haeupler, and Zvi Lotker.
IEEE/ACM Transactions on Networking (**ToN**), to appear.

Online Balanced Repartitioning
Chen Avin, Andreas Loukas, Maciej Pacut, and Stefan Schmid.
ArXiv Technical Report, November 2015.

# Conclusion

- Virtualized and programmable networked systems introduce many flexibilities, also at runtime
  - Resource allocation
  - Replica selection
- But also challenges
  - Predictable performance vs resource sharing?
  - Algorithmic challenges: multi-dimensional and online
  - Deployment